ACL-IJCNLP 2009

Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing of the AFNLP

Proceedings of the Conference Short Papers

4 August 2009
Suntec, Singapore

Production and Manufacturing by
World Scientific Publishing Co Pte Ltd
5 Toh Tuck Link
Singapore 596224

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

Table of Contents

Conference Program

Tuesday, August 4, 2009

Session 4DI: Short Paper 1 (Syntax and Parsing)
Chaired by Joakim Nivre

8:30–8:45 *Variational Inference for Grammar Induction with Prior Knowledge*
Shay Cohen and Noah A Smith

8:45–9:00 *Bypassed alignment graph for learning coordination in Japanese sentences*
Hideharu Okuma, Kazuo Hara, Masashi Shimbo and Yuji Matsumoto

9:00–9:15 *An Earley Parsing Algorithm for Range Concatenation Grammars*
Laura Kallmeyer, Wolfgang Maier and Yannick Parmentier

Session 4DII: Short Paper 2 (Discourse and Dialogue)
Chaired by Dragomir Radev

9:15–9:30 *Using Syntax to Disambiguate Explicit Discourse Connectives in Text*
Emily Pitler and Ani Nenkova

9:30–9:45 *Hybrid Approach to User Intention Modeling for Dialog Simulation*
Sangkeun Jung, Cheongjae Lee, Kyungduk Kim and Gary Geunbae Lee

12:20–14:20 Short Paper Poster Session
Chaired by Pushpak Bhattacharya, Virach Sornlertlamvanich

Cluster 1: Phonology, Word Segmentation and POS Tagging (P1-4)

Homophones and Tonal Patterns in English-Chinese Transliteration
Oi Yee Kwong

Capturing Errors in Written Chinese Words
Chao-Lin Liu, Kan-Wen Tien, Min-Hua Lai, Yi-Hsuan Chuang and Shih-Hung Wu

A Novel Word Segmentation Approach for Written Languages with Word Boundary Markers
Han-Cheol Cho, Do-Gil Lee, Jung-Tae Lee, Pontus Stenetorp, Jun'ichi Tsujii and Hae-Chang Rim

Part of Speech Tagger for Assamese Text
Navanath Saharia, Dhrubajyoti Das, Utpal Sharma and Jugal Kalita

Tuesday, August 4, 2009 (continued)

Cluster 2: Syntax and Parsing (P5-9)

Improving data-driven dependency parsing using large-scale LFG grammars
Lilja Øvrelid, Jonas Kuhn and Kathrin Spreyer

Incremental Parsing with Monotonic Adjoining Operation
Yoshihide Kato and Shigeki Matsubara

Bayesian Learning of a Tree Substitution Grammar
Matt Post and Daniel Gildea

A Unified Single Scan Algorithm for Japanese Base Phrase Chunking and Dependency Parsing
Manabu Sassano and Sadao Kurohashi

Comparing the Accuracy of CCG and Penn Treebank Parsers
Stephen Clark and James R. Curran

Cluster 3: Semantics (P10-14)

A Framework for Entailed Relation Recognition
Dan Roth, Mark Sammons and V.G.Vinod Vydiswaran

A Combination of Active Learning and Semi-supervised Learning Starting with Positive and Unlabeled Examples for Word Sense Disambiguation: An Empirical Study on Japanese Web Search Query
Makoto Imamura, Yasuhiro Takayama, Nobuhiro Kaji, Masashi Toyoda and Masaru Kitsuregawa

Detecting Compositionality in Multi-Word Expressions.
Ioannis Korkontzelos and Suresh Manandhar

Directional Distributional Similarity for Lexical Expansion
Lili Kotlerman, Ido Dagan, Idan Szpektor and Maayan Zhitomirsky-Geffet

Generalizing over Lexical Features: Selectional Preferences for Semantic Role Classification
Beñat Zapirain, Eneko Agirre and Lluís Màrquez

Tuesday, August 4, 2009 (continued)

Cluster 4: Discourse and Dialogue (P15-19)

A Syntactic and Lexical-Based Discourse Segmenter
Milan Tofiloski, Julian Brooke and Maite Taboada

Realistic Grammar Error Simulation using Markov Logic
Sungjin Lee and Gary Geunbae Lee

Discriminative Approach to Predicate-Argument Structure Analysis with Zero-Anaphora Resolution
Kenji Imamura, Kuniko Saito and Tomoko Izumi

Predicting Barge-in Utterance Errors by using Implicitly-Supervised ASR Accuracy and Barge-in Rate per User
Kazunori Komatani and Alexander I. Rudnicky

Automatic Generation of Information-seeking Questions Using Concept Clusters
Shuguang Li and Suresh Manandhar

Cluster 5: Summarization and Generation (P20-25)

Correlating Human and Automatic Evaluation of a German Surface Realiser
Aoife Cahill

Leveraging Structural Relations for Fluent Compressions at Multiple Compression Rates
Sourish Chaudhuri, Naman K. Gupta, Noah A. Smith and Carolyn P. Rose

Query-Focused Summaries or Query-Biased Summaries?
Rahul Katragadda and Vasudeva Varma

Using Generation for Grammar Analysis and Error Detection
Michael Goodman and Francis Bond

An Integrated Multi-document Summarization Approach based on Word Hierarchical Representation
You Ouyang, Wenjie Li and Qin Lu

Co-Feedback Ranking for Query-Focused Summarization
Furu Wei, Wenjie Li and Yanxiang He

Tuesday, August 4, 2009 (continued)

Cluster 6: Machine Translation (P26-32)

Reducing SMT Rule Table with Monolingual Key Phrase
Zhongjun He, Yao Meng, Yajuan Lü, Hao Yu and Qun Liu

A Statistical Machine Translation Model Based on a Synthetic Synchronous Grammar
Hongfei Jiang, Muyun Yang, Tiejun Zhao, Sheng Li and Bo Wang

English-Chinese Bi-Directional OOV Translation based on Web Mining and Supervised Learning
Yuejie Zhang, Yang Wang and Xiangyang Xue

The Backtranslation Score: Automatic MT Evalution at the Sentence Level without Reference Translations
Reinhard Rapp

Sub-Sentence Division for Tree-Based Machine Translation
Hao Xiong, Wenwen Xu, Haitao Mi, Yang Liu and Qun Liu

Asynchronous Binarization for Synchronous Grammars
John DeNero, Adam Pauls and Dan Klein

Hidden Markov Tree Model in Dependency-based Machine Translation
Zdenek Zabokrtsky and Martin Popel

Cluster 7: Sentiment Analysis (P33-40)

Word to Sentence Level Emotion Tagging for Bengali Blogs
Dipankar Das and Sivaji Bandyopadhyay

Extracting Comparative Sentences from Korean Text Documents Using Comparative Lexical Patterns and Machine Learning Techniques
Seon Yang and Youngjoong Ko

Opinion and Generic Question Answering Systems: a Performance Analysis
Alexandra Balahur, Ester Boldrini, Andrés Montoyo and Patricio Martínez-Barco

Automatic Satire Detection: Are You Having a Laugh?
Clint Burfoot and Timothy Baldwin

Tuesday, August 4, 2009 (continued)

Cluster 9: Text Mining and NLP Applications (P45-47)

Extracting Paraphrases of Technical Terms from Noisy Parallel Software Corpora
Xiaoyin Wang, David Lo, Jing Jiang, Lu Zhang and Hong Mei

Mining Association Language Patterns for Negative Life Event Classification
Liang-Chih Yu, Chien-Lung Chan, Chung-Hsien Wu and Chao-Cheng Lin

Automatic Compilation of Travel Information from Automatically Identified Travel Blogs
Hidetsugu Nanba, Haruka Taguma, Takahiro Ozaki, Daisuke Kobayashi, Aya Ishino and Toshiyuki Takezawa

Cluster 10: Language Resources (P48-51)

Play the Language: Play Coreference
Barbora Hladká, Jiří Mírovský and Pavel Schlesinger

Chinese Term Extraction Using Different Types of Relevance
Yuhang Yang, Tiejun Zhao, Qin Lu, Dequan Zheng and Hao Yu

iChi: a bilingual dictionary generating tool
István Varga and Shoichi Yokoyama

CATiB: The Columbia Arabic Treebank
Nizar Habash and Ryan Roth

Session 6A: Short Paper 3 (Machine Translation)
Chaired by Philipp Koehn

14:20–14:35 *A Beam-Search Extraction Algorithm for Comparable Data*
Christoph Tillmann

14:35–14:50 *Optimizing Word Alignment Combination For Phrase Table Training*
Yonggang Deng and Bowen Zhou

14:50–15:05 *Bridging Morpho-Syntactic Gap between Source and Target Sentences for English-Korean Statistical Machine Translation*
Gumwon Hong, Seung-Wook Lee and Hae-Chang Rim

Tuesday, August 4, 2009 (continued)

15:05–15:20 *Toward Smaller, Faster, and Better Hierarchical Phrase-based SMT*
Mei Yang and Jing Zheng

15:20–15:35 *Handling phrase reorderings for machine translation*
Yizhao Ni, Craig Saunders, Sandor Szedmak and Mahesan Niranjan

Session 6B: Short Paper 4 (Semantics)
Chaired by Dekang Lin

14:20–14:35 *Syntax is from Mars while Semantics from Venus! Insights from Spectral Analysis of Distributional Similarity Networks*
Chris Biemann, Monojit Choudhury and Animesh Mukherjee

14:35–14:50 *Introduction of a new paraphrase generation tool based on Monte-Carlo sampling*
Jonathan Chevelu, Thomas Lavergne, Yves Lepage and Thierry Moudenc

14:50–15:05 *Prediction of Thematic Rank for Structured Semantic Role Labeling*
Weiwei Sun, Zhifang Sui and Meng Wang

15:05–15:20 *Transfer Learning, Feature Selection and Word Sense Disambiguation*
Paramveer S. Dhillon and Lyle H. Ungar

Session 6C: Short Paper 5 (Spoken Language Processing)
Chaired by Sanjeev Khudanpur

14:20–14:35 *From Extractive to Abstractive Meeting Summaries: Can It Be Done by Sentence Compression?*
Fei Liu and Yang Liu

14:35–14:50 *Automatic Story Segmentation using a Bayesian Decision Framework for Statistical Models of Lexical Chain Features*
Wai-Kit Lo, Wenying Xiong and Helen Meng

14:50–15:05 *Investigating Pitch Accent Recognition in Non-native Speech*
Gina-Anne Levow

15:05–15:20 *A Stochastic Finite-State Morphological Parser for Turkish*
Haşim Sak, Tunga Güngör and Murat Saraçlar

15:20–15:35 *Parsing Speech Repair without Specialized Grammar Symbols*
Tim Miller, Luan Nguyen and William Schuler

Tuesday, August 4, 2009 (continued)

Session 6D: Short Paper 6 (Statistical and Machine Learning Methods 1)
Chaired by Yuji Mastumoto

14:20–14:35 *Efficient Inference of CRFs for Large-Scale Natural Language Data*
Minwoo Jeong, Chin-Yew Lin and Gary Geunbae Lee

14:35–14:50 *Iterative Scaling and Coordinate Descent Methods for Maximum Entropy*
Fang-Lan Huang, Cho-Jui Hsieh, Kai-Wei Chang and Chih-Jen Lin

14:50–15:05 *Automatic Cost Estimation for Tree Edit Distance Using Particle Swarm Optimization*
Yashar Mehdad

15:05–15:20 *Markov Random Topic Fields*
Hal Daume III

Session 6E: Short Paper 7 (Summarization and Generation)
Chaired by Diana Inkpen

15:40–15:55 *Multi-Document Summarization using Sentence-based Topic Models*
Dingding Wang, Shenghuo Zhu, Tao Li and Yihong Gong

15:55–16:10 *Validating the web-based evaluation of NLG systems*
Alexander Koller, Kristina Striegnitz, Donna Byron, Justine Cassell, Robert Dale, Sara Dalzel-Job, Johanna Moore and Jon Oberlander

16:10–16:25 *Extending a Surface Realizer to Generate Coherent Discourse*
Eva Banik

Session 6F: Short Paper 8 (Sentiment Analysis)
Chaired by Bo Pang

15:25–15:40 *The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language*
Rada Mihalcea and Carlo Strapparava

15:40–15:55 *Generalizing Dependency Features for Opinion Mining*
Mahesh Joshi and Carolyn Penstein-Rosé

15:55–16:10 *Graph Ranking for Sentiment Transfer*
Qiong Wu, Songbo Tan and Xueqi Cheng

Tuesday, August 4, 2009 (continued)

16:10–16:25 *The Contribution of Stylistic Information to Content-based Mobile Spam Filtering*
Dae-Neung Sohn, Jung-Tae Lee and Hae-Chang Rim

Session 6G: Short Paper 9 (Question Answering)
Chaired by Tomek Strzalkowski

15:40–15:55 *Learning foci for Question Answering over Topic Maps*
Alexander Mikhailian, Tiphaine Dalmas and Rani Pinchuk

15:55–16:10 *Do Automatic Annotation Techniques Have Any Impact on Supervised Complex Question Answering?*
Yllias Chali, Sadid Hasan and Shafiq Joty

16:10–16:25 *Where's the Verb? Correcting Machine Translation During Question Answering*
Wei-Yun Ma and Kathy McKeown

Session 6H: Short Paper 10 (Statistical and Machine Learning Methods 2)
Chaired by Kenneth Church

15:25–15:40 *A Note on the Implementation of Hierarchical Dirichlet Processes*
Phil Blunsom, Trevor Cohn, Sharon Goldwater and Mark Johnson

15:40–15:55 *A Succinct N-gram Language Model*
Taro Watanabe, Hajime Tsukada and Hideki Isozaki

15:55–16:10 *Modeling Morphologically Rich Languages Using Split Words and Unstructured Dependencies*
Deniz Yuret and Ergun Bicici

16:10–16:25 *Improved Smoothing for N-gram Language Models Based on Ordinary Counts*
Robert C. Moore and Chris Quirk

Tuesday, August 4, 2009 (continued)

Session 7D: Short Paper 11 (Information Extraction)
Chaired by Raymond Mooney

16:50–17:05 *Updating a Name Tagger Using Contemporary Unlabeled Data*
Cristina Mota and Ralph Grishman

17:05–17:20 *Arabic Cross-Document Coreference Resolution*
Asad Sayeed, Tamer Elsayed, Nikesh Garera, David Alexander, Tan Xu, Doug Oard, David Yarowsky and Christine Piatko

17:20–17:35 *The Impact of Query Refinement in the Web People Search Task*
Javier Artiles, Julio Gonzalo and Enrique Amigó

17:35–17:50 *Composite Kernels For Relation Extraction*
Frank Reichartz, Hannes Korte and Gerhard Paass

17:50–18:05 *Predicting Unknown Time Arguments based on Cross-Event Propagation*
Prashant Gupta and Heng Ji

Variational Inference for Grammar Induction with Prior Knowledge

Shay B. Cohen and **Noah A. Smith**
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{scohen,nasmith}@cs.cmu.edu

Abstract

Variational EM has become a popular technique in probabilistic NLP with hidden variables. Commonly, for computational tractability, we make strong independence assumptions, such as the mean-field assumption, in approximating posterior distributions over hidden variables. We show how a looser restriction on the approximate posterior, requiring it to be a mixture, can help inject prior knowledge to exploit soft constraints during the variational E-step.

1 Introduction

Learning natural language in an unsupervised way commonly involves the expectation-maximization (EM) algorithm to optimize the parameters of a generative model, often a probabilistic grammar (Pereira and Schabes, 1992). Later approaches include variational EM in a Bayesian setting (Beal and Gharamani, 2003), which has been shown to obtain even better results for various natural language tasks over EM (e.g., Cohen et al., 2008).

Variational EM usually makes the mean-field assumption, factoring the posterior over hidden variables into independent distributions. Bishop et al. (1998) showed how to use a less strict assumption: a *mixture* of factorized distributions.

In other work, soft or hard constraints on the posterior during the E-step have been explored in order to improve performance. For example, Smith and Eisner (2006) have penalized the approximate posterior over dependency structures in a natural language grammar induction task to avoid long range dependencies between words. Graça et al. (2007) added linear constraints on expected values of features of the hidden variables in an alignment task.

In this paper, we use posterior mixtures to inject bias or prior knowledge into a Bayesian model. We show that empirically, injecting prior knowledge improves performance on an unsupervised Chinese grammar induction task.

2 Variational Mixtures with Constraints

Our EM variant encodes prior knowledge in an approximate posterior by constraining it to be from a *mixture* family of distributions. We will use $\mathbf{x}$ to denote observable random variables, $\mathbf{y}$ to denote hidden structure, and $\boldsymbol{\theta}$ to denote the to-be-learned parameters of the model (coming from a subset of $\mathbb{R}^\ell$ for some ℓ). $\boldsymbol{\alpha}$ will denote the parameters of a prior over $\boldsymbol{\theta}$. The mean-field assumption in the Bayesian setting assumes that the posterior has a factored form:

$$q(\boldsymbol{\theta}, \mathbf{y}) = q(\boldsymbol{\theta})q(\mathbf{y}) \tag{1}$$

Traditionally, variational inference with the mean-field assumption alternates between an E-step which optimizes $q(\mathbf{y})$ and then an M-step which optimizes $q(\boldsymbol{\theta})$.[1] The mean-field assumption makes inference feasible, at the expense of optimizing a looser lower bound on the likelihood (Bishop, 2006). The lower bound that the algorithm optimizes is the following:

$$F(q(\boldsymbol{\theta}, \mathbf{y}), \boldsymbol{\alpha}) = \mathbb{E}_{q(\boldsymbol{\theta},\mathbf{y})}[\log p(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta} \mid \alpha)] + H(q) \tag{2}$$

where $H(q)$ denotes the entropy of distribution q. We focus on changing the E-step and as a result, changing the underlying bound, $F(q(\boldsymbol{\theta}, \mathbf{y}), \boldsymbol{\alpha})$. Similarly to Bishop et al. (1998), instead of making the strict mean-field assumption, we assume that the variational model is a mixture. One component of the mixture might take the traditional form, but others will be used to encourage certain

[1] This optimization can be nested inside another EM algorithm that optimizes $\boldsymbol{\alpha}$; this is our approach. $q(\boldsymbol{\theta})$ is traditionally conjugate to the likelihood for computational reasons, but our method is not limited to that kind of prior, as seen in the experiments.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 1–4,
Suntec, Singapore, 4 August 2009.

tendencies considered *a priori* to be appropriate. Denoting the probability simplex of dimension r $\triangle_r = \{\langle \lambda_1, ..., \lambda_r \rangle \in \mathbb{R}^r : \lambda_i \geq 0, \sum_{i=1}^r \lambda_i = 1\}$, we require that:

$$q(\boldsymbol{\theta}, \mathbf{y} \mid \boldsymbol{\lambda}) \quad = \quad \sum_{i=1}^r \lambda_i q_i(\mathbf{y}) q_i(\boldsymbol{\theta}) \qquad (3)$$

for $\boldsymbol{\lambda} \in \triangle_r$. $\mathcal{Q}_i$ will denote the family of distributions for the ith mixture component, and $\mathcal{Q}(\triangle_r)$ will denote the family implied by the mixture of $\mathcal{Q}_1, \ldots, \mathcal{Q}_r$ where the mixture coefficients $\boldsymbol{\lambda} \in \triangle_r$. $\boldsymbol{\lambda}$ comprise r additional variational parameters, in addition to parameters for each $q_i(\mathbf{y})$ and $q_i(\boldsymbol{\theta})$.

When one of the mixture components q_i is sufficiently expressive, $\boldsymbol{\lambda}$ will tend toward a degenerate solution. In order to force all mixture components to play a role—even at the expense of the tightness of the variational bound—we will impose hard constraints on $\boldsymbol{\lambda}$: $\boldsymbol{\lambda} \in \tilde{\triangle}_r \subset \triangle_r$. In our experiments (§3), $\tilde{\triangle}_r$ will be mostly a line segment corresponding to two mixture coefficients.

The role of the variational EM algorithm is to optimize the variational bound in Eq. 2 with respect to $q(\mathbf{y})$, $q(\boldsymbol{\theta})$, and $\boldsymbol{\lambda}$. Keeping this intention in mind, we can replace the E-step and M-step in the original variational EM algorithm with $2r + 1$ coordinate ascent steps, for $1 \leq i \leq r$:

E-step: For each $i \in \{1, ..., r\}$, optimize the bound given $\boldsymbol{\lambda}$ and $q_{i'}(\mathbf{y})|_{i' \in \{1,...,r\} \setminus \{i\}}$ and $q_{i'}(\boldsymbol{\theta})|_{i' \in \{1,...,r\}}$ by selecting a new distribution $q_i(\mathbf{y})$.

M-step: For each $i \in \{1, ..., r\}$, optimize the bound given $\boldsymbol{\lambda}$ and $q_{i'}(\boldsymbol{\theta})|_{i' \in \{1,...,r\} \setminus \{i\}}$ and $q_{i'}(\mathbf{y})|_{i' \in \{1,...,r\}}$ by selecting a new distribution $q_i(\boldsymbol{\theta})$.

C-step: Optimize the bound by selecting a new set of coefficients $\boldsymbol{\lambda} \in \tilde{\triangle}_r$ in order to optimize the bound with respect to the mixture coefficients.

We call the revised algorithm **constrained mixture variational EM**.

For a distribution $r(\mathbf{h})$, we denote by $\mathrm{KL}(\mathcal{Q}_i \| r)$ the following:

$$\mathrm{KL}(\mathcal{Q}_i \| r) = \min_{q \in \mathcal{Q}_i} \mathrm{KL}(q(\mathbf{h}) \| r)) \qquad (4)$$

where $\mathrm{KL}(\cdot \| \cdot)$ denotes the Kullback-Leibler divergence.

The next proposition, which is based on a result in Graça et al. (2007), gives an intuition of how modifying the variational EM algorithm with $\mathcal{Q} = \mathcal{Q}(\tilde{\triangle}_r)$ affects the solution:

Proposition 1. *Constrained mixture variational EM finds local maxima for a function $G(q, \boldsymbol{\alpha})$ such that*

$$\log p(x \mid \boldsymbol{\alpha}) - \min_{\boldsymbol{\lambda} \in \tilde{\triangle}_r} L(\boldsymbol{\lambda}, \boldsymbol{\alpha}) \leq G(q, \boldsymbol{\alpha}) \leq \log p(x \mid \boldsymbol{\alpha}) \qquad (5)$$

where $L(\boldsymbol{\lambda}, \boldsymbol{\alpha}) = \sum_{i=1}^r \lambda_i \mathrm{KL}(\mathcal{Q}_i \| p(\boldsymbol{\theta}, \mathbf{y} \mid \mathbf{x}, \boldsymbol{\alpha}))$.

We can understand mixture variational EM as penalizing the likelihood with a term bounded by a linear function of the $\boldsymbol{\lambda}$, minimized over $\tilde{\triangle}_r$. We will exploit that bound in §2.2 for computational tractability.

2.1 Simplex Annealing

The variational EM algorithm still identifies only *local* maxima. Different proposals have been for pushing EM toward a global maximum. In many cases, these methods are based on choosing different initializations for the EM algorithm (e.g., repeated random initializations or a single carefully designed initializer) such that it eventually gets closer to a global maximum.

We follow the idea of annealing proposed in Rose et al. (1990) and Smith and Eisner (2006) for the $\boldsymbol{\lambda}$ by gradually loosening hard constraints on $\boldsymbol{\lambda}$ as the variational EM algorithm proceeds. We define a sequence of $\tilde{\triangle}_r(t)$ for $t = 0, 1, ...$ such that $\tilde{\triangle}_r(t) \subseteq \tilde{\triangle}_r(t+1)$. First, we have the inequality:

$$\begin{aligned} &\mathrm{KL}(\mathcal{Q}(\tilde{\triangle}_r(t)) \| p(\boldsymbol{\theta}, \mathbf{y} \mid \mathbf{x}, \boldsymbol{\alpha}) \\ &\quad \geq \quad \mathrm{KL}(\mathcal{Q}(\tilde{\triangle}_r(t+1)) \| p(\boldsymbol{\theta}, \mathbf{y} \mid \mathbf{x}, \boldsymbol{\alpha})) \end{aligned} \qquad (6)$$

We say that the annealing schedule is τ-separated if we have for any $\boldsymbol{\alpha}$:

$$\begin{aligned} &\mathrm{KL}(\mathcal{Q}(\tilde{\triangle}_r(t)) \| p(\boldsymbol{\theta}, \mathbf{y} \mid \mathbf{x}, \boldsymbol{\alpha})) \\ &\quad \leq \quad \mathrm{KL}(\mathcal{Q}(\tilde{\triangle}_r(t+1)) \| p(\boldsymbol{\theta}, \mathbf{y} \mid \mathbf{x}, \boldsymbol{\alpha})) - \frac{\tau}{2^{(t+1)}} \end{aligned} \qquad (7)$$

τ-separation requires consecutive families $\mathcal{Q}(\tilde{\triangle}_r(t))$ and $\mathcal{Q}(\tilde{\triangle}_r(t+1))$ to be similar.

Proposition 1 stated the bound we optimize, which penalizes the likelihood by subtracting a positive KL divergence from it. With the τ-separation condition we can show that even though we penalize likelihood, the variational EM algorithm will still increase likelihood by a certain amount. Full details are omitted for space and can be found in **?**).

Input: initial parameters $\boldsymbol{\alpha}^{(0)}$, observed data $\mathbf{x}$, annealing schedule $\tilde{\triangle}_r : \mathbb{N} \rightarrow 2^{\triangle_r}$
Output: learned parameters $\boldsymbol{\alpha}$ and approximate posterior $q(\boldsymbol{\theta}, \mathbf{y})$

$t \leftarrow 1$;
repeat
  E-step: **repeat**
    E-step: **forall** $i \in [r]$ do: $q_i^{(t+1)}(\mathbf{y}) \leftarrow \operatorname{argmax}_{q(\mathbf{y}) \in \mathcal{Q}_i} F'(\sum_{j \neq i} \lambda_j q_i^{(t)}(\boldsymbol{\theta}) q(\mathbf{y}) + \lambda_i q_i^{(t)} q(\mathbf{y}), \boldsymbol{\alpha}^{(t)})$
    M-step: **forall** $i \in [r]$ do: $q_i^{(t+1)}(\boldsymbol{\theta}) \leftarrow \operatorname{argmax}_{q(\boldsymbol{\theta}) \in \mathcal{Q}_i} F'(\sum_{j \neq i} \lambda_j q(\boldsymbol{\theta}) q_i^{(t)}(\mathbf{y}) + \lambda_i q_i^{(t)} q(\mathbf{y}), \boldsymbol{\alpha}^{(t)})$
    C-step: $\boldsymbol{\lambda}^{(t+1)} \leftarrow \operatorname{argmax}_{\boldsymbol{\lambda} \in \tilde{\triangle}_r(t)} F'(\sum_{j=1}^{r} \lambda_j q_i^{(t)}(\boldsymbol{\theta}) q_i^{(t)}(\mathbf{y}), \boldsymbol{\alpha}^{(t)})$
  until *convergence* ;
  M-step: $\boldsymbol{\alpha}^{(t+1)} \leftarrow \operatorname{argmax}_{\boldsymbol{\alpha}} F'(\sum_{i=1}^{r} \lambda_i q_i^{(t+1)}(\boldsymbol{\theta}) q_i^{(t+1)}(\mathbf{y}), \boldsymbol{\alpha})$
  $t \leftarrow t + 1$;
until *convergence* ;
return $\boldsymbol{\alpha}^{(t)}, \sum_{i=1}^{r} \lambda_i q_i^{(t)}(\boldsymbol{\theta}) q_i^{(t)}(\mathbf{y})$

Figure 1: The constrained variational mixture EM algorithm. $[n]$ denotes $\{1, ..., n\}$.

2.2 Tractability

We now turn to further alterations of the bound in Eq. 2 to make it more tractable. The main problem is the entropy term which is not easy to compute, because it includes a $\log$ term over a mixture of distributions from $\mathcal{Q}_i$. We require the distributions in $\mathcal{Q}_i$ to factorize over the hidden structure $\mathbf{y}$, but this only helps with the first term in Eq. 2.

We note that because the entropy function is convex, we can get a lower bound on $H(q)$:

$$H(q) \geq \sum_{i=1}^{r} \lambda_i H(q_i) = \sum_{i=1}^{r} \lambda_i H(q_i(\boldsymbol{\theta}, \mathbf{y}))$$

Substituting the modified entropy term into Eq. 2 still yields a lower bound on the likelihood. This change makes the E-step tractable, because each distribution $q_i(\mathbf{y})$ can be computed separately by optimizing a bound which depends only on the variational parameters in that distribution. In fact, the bound on the left hand side in Proposition 1 becomes the function that we optimize instead of $G(q, \boldsymbol{\alpha})$.

Without proper constraints, the $\boldsymbol{\lambda}$ update can be intractable as well. It requires maximizing a linear objective (in $\boldsymbol{\lambda}$) while constraining the $\boldsymbol{\lambda}$ to be from a particular subspace of the probability simplex, $\tilde{\triangle}_r(t)$. To solve this issue, we require that $\tilde{\triangle}_r(t)$ is polyhedral, making it possible to apply linear programming (Boyd and Vandenberghe, 2004).

The bound we optimize is:[2]

$$F'\left(\sum_{i=1}^{r} \lambda_i q_i(\boldsymbol{\theta}, \mathbf{y}), \boldsymbol{\alpha}\right) \tag{8}$$
$$= \sum_{i=1}^{r} \lambda_i \left(\mathbb{E}_{q_i(\boldsymbol{\theta}, \mathbf{y})}[\log p(\boldsymbol{\theta}, \mathbf{y}, \mathbf{x} \mid \mathbf{m})] + H(q_i(\boldsymbol{\theta}, \mathbf{y}))\right)$$

with $\boldsymbol{\lambda} \in \tilde{\triangle}_r(t_{\text{final}})$ and $(q_i(\boldsymbol{\theta}, \mathbf{y})) \in \mathcal{Q}_i$. The algorithm for optimizing this bound is in Fig. 1, which includes an extra M-step to optimize $\boldsymbol{\alpha}$ (see extended report).

3 Experiments

We tested our method on the unsupervised learning problem of dependency grammar induction. For the generative model, we used the *dependency model with valence* as it appears in Klein and Manning (2004). We used the data from the Chinese treebank (Xue et al., 2004). Following standard practice, sentences were stripped of words and punctuation, leaving part-of-speech tags for the unsupervised induction of dependency structure, and sentences of length more than 10 were removed from the set. We experimented with a Dirichlet prior over the parameters and logistic normal priors over the parameters, and found the latter to still be favorable with our method, as in Cohen et al. (2008). We therefore report results with our method only for the logistic normal prior. We do inference on sections 1–270 and 301–1151 of CTB10 (4,909 sentences) by running the EM algorithm for 20 iterations, for which all algorithms have their variational bound converge.

To evaluate performance, we report the fraction of words whose predicted parent matches the gold standard (attachment accuracy). For parsing, we use the minimum Bayes risk parse.

Our mixture components $\mathcal{Q}_i$ are based on simple linguistic tendencies of Chinese syntax. These observations include the tendency of dependencies to (a) emanate from the right of the current position and (b) connect words which are nearby (in string distance). We experiment with six mixture components: (1) RIGHTATTACH: Each word's parent is to the word's right. The root, therefore, is always the rightmost word; (2) ALLRIGHT: The rightmost word is the parent of all positions in the sentence (there is only one such tree); (3) LEFTCHAIN: The tree forms a chain, such that each

[2] This is a less tight bound than the one in Bishop et al. (1998), but it is easier to handle computationally.

learning setting				
	LEFTCHAIN	34.9		
	vanilla EM	38.3		
	LN, mean-field	48.9		
	This paper:	I	II	III
	RIGHTATTACH	49.1	47.1	**49.8**
	ALLRIGHT	**49.4**	**49.4**	48.4
	LEFTCHAIN	47.9	46.5	**49.9**
	VERBASROOT	**50.5**	50.2	49.4
	NOUNSEQUENCE	48.9	48.9	**49.9**
	SHORTDEP	**49.5**	48.4	48.4
	RA+VAR+SD	50.5	**50.6**	50.1

Table 1: Results (attachment accuracy). The baselines are LEFTCHAIN as a parsing model (attaches each word to the word on its right), non-Bayesian EM, and mean-field variational EM without any constraints. These are compared against the six mixture components mentioned in the text. (I) corresponds to simplex annealing experiments ($\lambda_1^{(0)} = 0.85$); (II–III) correspond to fixed values, 0.85 and 0.95, for the mixture coefficients. With the last row, λ_2 to λ_4 are always $(1 - \lambda_1)/3$. Boldface denotes the best result in each row.

word is governed by the word to its right; (4) VERBASROOT: Only verbs can attach to the wall node \$; (5) NOUNSEQUENCE: Every sequence of n NN (nouns) is assumed to be a noun phrase, hence the first $n-1$ NNs are attached to the last NN; and (6) SHORTDEP: Allow only dependencies of length four or less. This is a strict model reminiscent of the successful application of *structural bias* to grammar induction (Smith and Eisner, 2006).

These components are added to a variational DMV model without the sum-to-1 constraint on $\boldsymbol{\theta}$. This complements variational techniques which state that the optimal solution during the E-step for the mean-field variational EM algorithm is a weighted grammar of the same form of $p(\mathbf{x}, \mathbf{y} \mid \boldsymbol{\theta})$ (DMV in our case). Using the mixture components this way has the effect of *smoothing* the estimated grammar event counts during the E-step, in the direction of some prior expectations.

Let λ_1 correspond to the component of the original DMV model, and let λ_2 correspond to one of the components from the above list. Variational techniques show that if we let λ_1 obtain the value 1, then the optimal solution will be $\lambda_1 = 1$ and $\lambda_2 = 0$. We therefore restrict λ_1 to be smaller than 1. More specifically, we use an annealing process which starts by limiting λ_1 to be $\leq s = 0.85$ (and hence limits λ_2 to be ≥ 0.15) and increases s at each step by 1% until s reaches 0.95. In addition, we also ran the algorithm with λ_1 fixed at 0.85 and λ_1 fixed at 0.95 to check the effectiveness of annealing on the simplex.

Table 1 describes the results of our experiments. In general, using additional mixture components has a clear advantage over the mean-field assumption. The best result with a single mixture is achieved with annealing, and the VERBASROOT component. A combination of the mixtures (RIGHTATTACH) together with VERBASROOT and SHORTDEP led to an additional improvement, implying that proper selection of several mixture components together can achieve a performance gain.

4 Conclusion

We described a variational EM algorithm that uses a mixture model for the variational model. We refined the algorithm with an annealing mechanism to avoid local maxima. We demonstrated the effectiveness of the algorithm on a dependency grammar induction task. Our results show that with a good choice of mixture components and annealing schedule, we achieve improvements for this task over mean-field variational inference.

References

M. J. Beal and Z. Gharamani. 2003. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In *Proc. of Bayesian Statistics*.

C. Bishop, N. Lawrence, T. S. Jaakkola, and M. I. Jordan. 1998. Approximating posterior distributions in belief networks using mixtures. In *Advances in NIPS*.

C. M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

S. Boyd and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge Press.

S. B. Cohen and N. A. Smith. 2009. Variational inference with prior knowledge. Technical report, Carnegie Mellon University.

S. B. Cohen, K. Gimpel, and N. A. Smith. 2008. Logistic normal priors for unsupervised probabilistic grammar induction. In *Advances in NIPS*.

J. V. Graça, K. Ganchev, and B. Taskar. 2007. Expectation maximization and posterior constraints. In *Advances in NIPS*.

D. Klein and C. D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*.

F. C. N. Pereira and Y. Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proc. of ACL*.

K. Rose, E. Gurewitz, and G. C. Fox. 1990. Statistical mechanics and phrase transitions in clustering. *Physical Review Letters*, 65(8):945–948.

N. A. Smith and J. Eisner. 2006. Annealing structural bias in multilingual weighted grammar induction. In *Proc. of COLING-ACL*.

N. Xue, F. Xia, F.-D. Chiou, and M. Palmer. 2004. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 10(4):1–30.

Bypassed Alignment Graph for Learning Coordination in Japanese Sentences

Hideharu Okuma Kazuo Hara Masashi Shimbo Yuji Matsumoto
Graduate School of Information Science
Nara Institute of Science and Technology
Ikoma, Nara 630-0192, Japan
{okuma.hideharu01,kazuo-h,shimbo,matsu}@is.naist.jp

Abstract

Past work on English coordination has focused on coordination scope disambiguation. In Japanese, detecting whether coordination exists in a sentence is also a problem, and the state-of-the-art alignment-based method specialized for scope disambiguation does not perform well on Japanese sentences. To take the detection of coordination into account, this paper introduces a 'bypass' to the alignment graph used by this method, so as to explicitly represent the non-existence of coordinate structures in a sentence. We also present an effective feature decomposition scheme based on the distance between words in conjuncts.

1 Introduction

Coordination remains one of the challenging problems in natural language processing. One key characteristic of coordination explored in the past is the structural and semantic symmetry of conjuncts (Chantree et al., 2005; Hogan, 2007; Resnik, 1999). Recently, Shimbo and Hara (2007) proposed to use a large number of features to model this symmetry, and optimize the feature weights with perceptron training. These features are assigned to the arcs of the *alignment graph* (or *edit graph*) originally developed for biological sequence alignment.

Coordinate structure analysis involves two related but different tasks:

1. Detect the presence of coordinate structure in a sentence (or a phrase).

2. Disambiguate the scope of coordinations in the sentences/phrases detected in Task 1.

The studies on English coordination listed above are concerned mainly with scope disambiguation, reflecting the fact that detecting the presence of coordinations in a sentence (Task 1) is straightforward in English. Indeed, nearly 100% precision and recall can be achieved in Task 1 simply by pattern matching with a small number of coordination markers such as "and," "or," and "as well as".

In Japanese, on the other hand, detecting coordination is non-trivial. Many of the coordination markers in Japanese are ambiguous and do not always indicate the presence of coordinations. Compare sentences (1) and (2) below:

(1) *rondon* *to* *pari* *ni* *itta*
(London) (and) (Paris) (to) (went)
(I went to London and Paris)

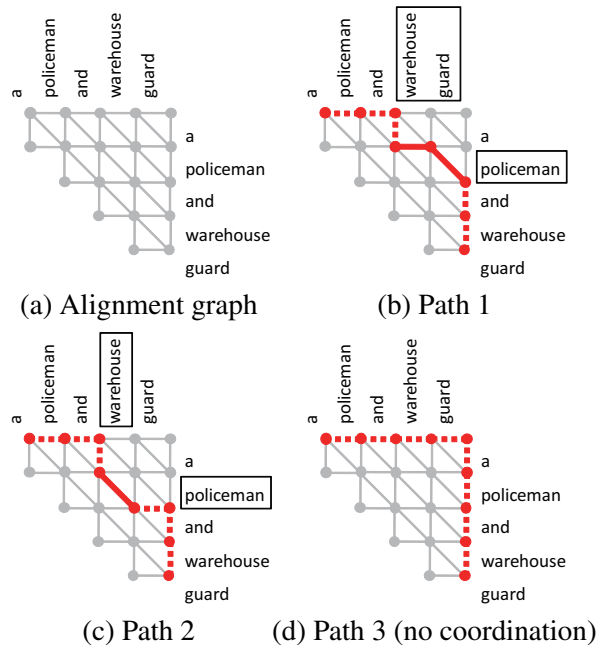
(2) *kanojo* *to* *pari* *ni* *itta*
(her) (with) (Paris) (to) (went)
(I went to Paris with her)

These sentences differ only in the first word. Both contain a particle *to*, which is one of the most frequent coordination markers in Japanese—but only the first sentence contains a coordinate structure. Pattern matching with particle *to* thus fails to filter out sentence (2).

Shimbo and Hara's model allows a sentence without coordinations to be represented as a normal path in the alignment graph, and in theory it can cope with Task 1 (detection). In practice, the representation is inadequate when a large number of training sentences do not contain coordinations, as demonstrated in the experiments of Section 4.

This paper presents simple yet effective modifications to the Shimbo-Hara model to take coordination detection into account, and solve Tasks 1 and 2 simultaneously.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 5–8,
Suntec, Singapore, 4 August 2009.



(a) Alignment graph (b) Path 1

(c) Path 2 (d) Path 3 (no coordination)

Figure 1: Alignment graph for "a policeman and warehouse guard" ((a)), and example paths representing different coordinate structure ((b)–(d)).

2 Alignment-based coordinate structure analysis

We first describe Shimbo and Hara's method upon which our improvements are made.

2.1 Triangular alignment graph

The basis of their method is a triangular *alignment graph*, illustrated in Figure 1(a). Kurohashi and Nagao (1994) used a similar data structure in their rule-based method. Given an input sentence, the rows and columns of its alignment graph are associated with the words in the sentence. Unlike the alignment graph used in biological sequence alignment, the graph is triangular because the same sentence is associated with rows and columns. Three types of arcs are present in the graph. A diagonal arc denotes coordination between the word above the arc and the one on the right; the horizontal and vertical arcs represent skipping of respective words.

Coordinate structure in a sentence is represented by a complete path starting from the top-left (initial) node and arriving at the bottom-right (terminal) node in its alignment graph. Each arc in this path is labeled either *Inside* or *Outside* depending on whether its span is part of coordination or not; i.e., the horizontal and vertical spans of an *Inside* segment determine the scope of two conjuncts. Figure 1(b)–(d) depicts example paths. *Inside* and *Outside* arcs are depicted by solid and dotted lines, respectively. Figure 1(b) shows a path for coordination between "policeman" (vertical span of the *Inside* segment) and "warehouse guard" (horizontal span). Figure 1(c) is for "policeman" and "warehouse." Non-existence of coordinations in a sentence is represented by the *Outside*-only path along the top and the rightmost borders of the graph (Figure 1(d)).

With this encoding of coordinations as paths, coordinate structure analysis can be reduced to finding the highest scoring path in the graph, where the score of an arc is given by a measure of how much two words are likely to be coordinated. The goal is to build a measure that assigns the highest score to paths denoting the correct coordinate structure. Shimbo and Hara defined this measure as a linear function of many features associated to arcs, and used perceptron training to optimize the weight coefficients for these features from corpora.

2.2 Features

For the description of features used in our adaptation of the Shimbo-Hara model to Japanese, see (Okuma et al., 2009). In this model, all features are defined as indicator functions asking whether one or more attributes (e.g., surface form, part-of-speech) take specific values at the neighbor of an arc. One example of a feature assigned to a diagonal arc at row i and column j of the alignment graph is

$$f = \begin{cases} 1 & \text{if } \mathit{POS}[i] = \text{Noun},\ \mathit{POS}[j] = \text{Adjective}, \\ & \text{and the label of the arc is } \textit{Inside}, \\ 0 & \text{otherwise.} \end{cases}$$

where $\mathit{POS}[i]$ denotes the part-of-speech of the ith word in a sentence.

3 Improvements

We introduce two modifications to improve the performance of Shimbo and Hara's model in Japanese coordinate structure analysis.

3.1 Bypassed alignment graphs

In their model, a path for a sentence with no coordination is represented as a series of *Outside* arcs as we saw in Figure 1(d). However, *Outside* arcs also appear in partial paths between two coordinations, as illustrated in Figure 2. Thus, two differ-

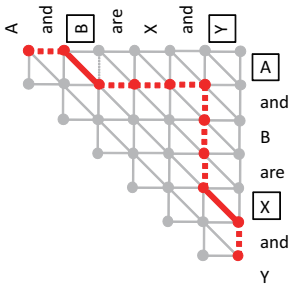


Figure 2: Original alignment graph for sentence with two coordinations. Notice that *Outside* (dotted) arcs connect two coordinations

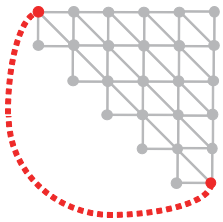


Figure 3: alignment graph with a "bypass"

ent roles are given to *Outside* arcs in the original Shimbo-Hara model.

We identify this to be a cause of their model not performing well for Japanese, and propose to augment the original alignment graph with a "bypass" devoted to explicitly indicate that no coordination exists in a sentence; i.e., we add a special path directly connecting the initial node and the terminal node of an alignment graph. See Figure 3 for illustration of a bypass.

In the new model, if the score of the path through the bypass is higher than that of any paths in the original alignment graph, the input sentence is deemed not containing coordinations.

We assign to the bypass two types of features capturing the characteristics of a whole sentence; i.e., indicator functions of sentence length, and of the existence of individual particles in a sentence. The weight of these features, which eventually determines the score of the bypass, is tuned by perceptron just like the weights of other features.

3.2 Making features dependent on the distance between conjuncts

Coordinations of different type (e.g., nominal and verbal) have different relevant features, as well as different average conjunct length (e.g., nominal coordinations are shorter).

This observation leads us to our second modification: to make all features dependent on their occurring positions in the alignment graph. To be precise, for each individual feature in the original model, a new feature is introduced which depends on whether the Manhattan distance d in the alignment graph between the position of the feature occurrence and the nearest diagonal exceeds a fixed threshold[1] θ. For instance, if a feature f is an indicator function of condition X, a new feature f' is introduced such that

$$f' = \begin{cases} 1, & \text{if } d \leq \theta \text{ and condition } X \text{ holds,} \\ 0, & \text{otherwise.} \end{cases}$$

Accordingly, different weights are learned and associated to two features f and f'. Notice that the Manhattan distance to the nearest diagonal is equal to the distance between word pairs to which the feature is assigned, which in turn is a rough estimate of the length of conjuncts.

This distance-based decomposition of features allows different feature weights to be learned for coordinations with conjuncts shorter than or equal to θ, and those which are longer.

4 Experimental setup

We applied our improved model and Shimbo and Hara's original model to the EDR corpus (EDR, 1995). We also ran the Kurohashi-Nagao parser (KNP) 2.0[2], a widely-used Japanese dependency parser to which Kurohashi and Nagao's (1994) rule-based coordination analysis method is built in. For comparison with KNP, we focus on *bunsetsu*-level coordinations. A bunsetsu is a chunk formed by a content word followed by zero or more non-content words like particles.

4.1 Dataset

The Encyclopedia section of the EDR corpus was used for evaluation. In this corpus, each sentence is segmented into words and is accompanied by a syntactic dependency tree, and a semantic frame representing semantic relations among words.

A coordination is indicated by a specific relation of type "and" in the semantic frame. The scope of conjuncts (where a conjunct may be a word, or a series of words) can be obtained by combining this information with that of the syntactic tree. The detail of this procedure can be found in (Okuma et al., 2009).

[1] We use $\theta = 5$ in the experiments of Section 4.

[2] http://nlp.kuee.kyoto-u.ac.jp/nl-resource/knp-e.html

Table 1: Accuracy of coordination scopes and end of conjuncts, averaged over five-fold cross validation. The numbers in brackets are the improvements (in points) relative to the Shimbo-Hara (SH) method.

	Scope of coordinations			End of conjuncts		
Method	Precision	Recall	F1 measure	Precision	Recall	F1 measure
KNP	n/a	n/a	n/a	58.8	65.3	61.9 (−2.6)
Shimbo and Hara's method (SH; baseline)	53.7	49.8	51.6 (±0.0)	67.0	62.1	64.5 (±0.0)
SH + distance-based feature decomposition	55.3	52.1	53.6 (+2.0)	68.3	64.3	66.2 (+1.7)
SH + distance-based feature decomposition + bypass	55.0	57.6	56.3 (+4.7)	66.8	69.9	68.3 (+3.8)

Of 10,072 sentences in the Encyclopedia section, 5,880 sentences contain coordinations. We excluded 1,791 sentences in which nested coordinations occur, as these cannot be processed with Shimbo and Hara's method (with or without our improvements).

We then applied Japanese morphological analyzer JUMAN 5.1 to segment each sentence into words and annotate them with parts-of-speech, and KNP with option '-bnst' to transform the series of words into a bunsetsu series. With this processing, each word-level coordination pair is also translated into a bunsetsu pair, unless the word-level pair is concatenated into a single bunsetsu (sub-bunsetsu coordination). Removing sub-bunsetsu coordinations and obvious annotation errors left us with 3,257 sentences with bunsetsu-level coordinations. Combined with the 4,192 sentences not containing coordinations, this amounts to 7,449 sentences used for our evaluation.

4.2 Evaluation metrics

KNP outputs dependency structures in Kyoto Corpus format (Kurohashi et al., 2000) which specifies the end of coordinating conjuncts (bunsetsu sequences) but not their beginning.

Hence two evaluation criteria were employed: (i) correctness of coordination scopes[3] (for comparison with Shimbo-Hara), and (ii) correctness of the end of conjuncts (for comparison with KNP). We report precision, recall and F1 measure, with the main performance index being F1 measure.

5 Results

Table 1 summarizes the experimental results. Even Shimbo and Hara's original method (SH) outperformed KNP. KNP tends to output too many coordinations, yielding a high recall but low precision. By contrast, SH outputs a smaller number of coordinations; this yields a high precision but a low recall.

The distance-based feature decomposition of Section 3.2 gave +2.0 points improvement over the original SH in terms of F1 measure in coordination scope detection. Adding bypasses to alignment graphs further improved the performance, making a total of +4.7 points in F1 over SH; recall significantly improved, with precision remaining mostly intact. Finally, the improved model (SH + decomposition + bypass) achieved an F1 measure +6.4 points higher than that of KNP in terms of end-of-conjunct identification.

[3] A coordination scope is deemed correct only if the bracketing of constituent conjuncts are all correct.

References

F. Chantree, A. Kilgarriff, A. de Roeck, and A. Willis. 2005. Disambiguating coordinations using word distribution information. In *Proc. 5th RANLP*.

EDR, 1995. *The EDR dictionary*. NICT. http://www2.nict.go.jp/r/r312/EDR/index.html.

D. Hogan. 2007. Coordinate noun phrase disambiguation in a generative parsing model. In *Proc. 45th ACL*, pages 680–687.

S. Kurohashi and M. Nagao. 1994. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Comput. Linguist.*, 20:507–534.

S. Kurohashi, Y. Igura, and M. Sakaguchi, 2000. *Annotation manual for a morphologically and sytactically tagged corpus, Ver. 1.8*. Kyoto Univ. In Japanese. http://nlp.kuee.kyoto-u.ac.jp/nl-resource/corpus/KyotoCorpus4.0/doc/syn_guideline.pdf.

H. Okuma, M. Shimbo, K. Hara, and Y. Matsumoto. 2009. Bypassed alignment graph for learning coordination in Japanese sentences: supplementary materials. Tech. report, Grad. School of Information Science, Nara Inst. Science and Technology. http://isw3.naist.jp/IS/TechReport/report-list.html#2009.

P. Resnik. 1999. Semantic similarity in a taxonomy. *J. Artif. Intel. Res.*, 11:95–130.

M. Shimbo and K. Hara. 2007. A discriminative learning model for coordinate conjunctions. In *Proc. 2007 EMNLP/CoNLL*, pages 610–619.

An Earley Parsing Algorithm for Range Concatenation Grammars

Laura Kallmeyer
SFB 441
Universität Tübingen
72074 Tübingen, Germany
lk@sfs.uni-tuebingen.de

Wolfgang Maier
SFB 441
Universität Tübingen
72074 Tübingen, Germany
wo.maier@uni-tuebingen.de

Yannick Parmentier
CNRS - LORIA
Nancy Université
54506 Vandœuvre, France
parmenti@loria.fr

Abstract

We present a CYK and an Earley-style algorithm for parsing Range Concatenation Grammar (RCG), using the deductive parsing framework. The characteristic property of the Earley parser is that we use a technique of range boundary constraint propagation to compute the yields of non-terminals as late as possible. Experiments show that, compared to previous approaches, the constraint propagation helps to considerably decrease the number of items in the chart.

1 Introduction

RCGs (Boullier, 2000) have recently received a growing interest in natural language processing (Søgaard, 2008; Sagot, 2005; Kallmeyer et al., 2008; Maier and Søgaard, 2008). RCGs generate exactly the class of languages parsable in deterministic polynomial time (Bertsch and Nederhof, 2001). They are in particular more powerful than linear context-free rewriting systems (LCFRS) (Vijay-Shanker et al., 1987). LCFRS is unable to describe certain natural language phenomena that RCGs actually can deal with. One example are long-distance scrambling phenomena (Becker et al., 1991; Becker et al., 1992). Other examples are non-semilinear constructions such as case stacking in Old Georgian (Michaelis and Kracht, 1996) and Chinese number names (Radzinski, 1991). Boullier (1999) shows that RCGs can describe the permutations occurring with scrambling and the construction of Chinese number names.

Parsing algorithms for RCG have been introduced by Boullier (2000), who presents a directional top-down parsing algorithm using pseudocode, and Barthélemy et al. (2001), who add an oracle to Boullier's algorithm. The more restricted class of LCFRS has received more attention concerning parsing (Villemonte de la Clergerie, 2002; Burden and Ljunglöf, 2005). This article proposes new CYK and Earley parsers for RCG, formulating them in the framework of parsing as deduction (Shieber et al., 1995). The second section introduces necessary definitions. Section 3 presents a CYK-style algorithm and Section 4 extends this with an Earley-style prediction.

2 Preliminaries

The rules (*clauses*) of RCGs[1] rewrite predicates ranging over parts of the input by other predicates. E.g., a clause $S(aXb) \rightarrow S(X)$ signifies that S is true for a part of the input if this part starts with an a, ends with a b, and if, furthermore, S is also true for the part between a and b.

Definition 1. *A* RCG $G = \langle N, T, V, P, S\rangle$ *consists of a) a finite set of predicates N with an arity function dim: $N \rightarrow \mathbb{N} \setminus \{0\}$ where $S \in N$ is the start predicate with $dim(S) = 1$, b) disjoint finite sets of terminals T and variables V, c) a finite set P of clauses $\psi_0 \rightarrow \psi_1 \dots \psi_m$, where $m \geq 0$ and each of the $\psi_i, 0 \leq i \leq m$, is a predicate of the form $A_i(\alpha_1, \dots, \alpha_{dim(A_i)})$ with $A_i \in N$ and $\alpha_j \in (T \cup V)^*$ for $1 \leq j \leq dim(A_i)$.*

Central to RCGs is the notion of ranges on strings.

Definition 2. *For every $w = w_1 \dots w_n$ with $w_i \in T$ $(1 \leq i \leq n)$, we define a) $Pos(w) = \{0, \dots, n\}$. b) $\langle l, r\rangle \in Pos(w) \times Pos(w)$ with $l \leq r$ is a* range *in w. Its* yield *$\langle l, r\rangle(w)$ is the substring $w_{l+1} \dots w_r$. c) For two ranges $\rho_1 = \langle l_1, r_1\rangle, \rho_2 = \langle l_2, r_2\rangle$: if $r_1 = l_2$, then $\rho_1 \cdot \rho_2 = \langle l_1, r_2\rangle$; otherwise $\rho_1 \cdot \rho_2$ is undefined. d) A vector $\phi = (\langle x_1, y_1\rangle, \dots, \langle x_k, y_k\rangle)$ is a* range vector *of dimension k in w if $\langle x_i, y_i\rangle$ is a range in w for $1 \leq i \leq k$. $\phi(i).l$ (resp. $\phi(i).r$) denotes then the*

[1] In this paper, by RCG, we always mean *positive RCG*, see Boullier (2000) for details.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 9–12,
Suntec, Singapore, 4 August 2009.

first (resp. second) component of the ith element of ϕ, that is x_i (resp. y_i).

In order to instantiate a clause of the grammar, we need to find ranges for all variables in the clause and for all occurrences of terminals. For convenience, we assume the variables in a clause and the occurrences of terminals to be equipped with distinct subscript indices, starting with 1 and ordered from left to right (where for variables, only the first occurrence is relevant for this order). We introduce a function $\Upsilon : P \to \mathbb{N}$ that gives the maximal index in a clause, and we define $\Upsilon(c, x)$ for a given clause c and x a variable or an occurrence of a terminal as the index of x in c.

Definition 3. *An* instantiation *of a $c \in P$ with $\Upsilon(c) = j$ w.r.t. to some string w is given by a range vector ϕ of dimension j. Applying ϕ to a predicate $A(\vec{\alpha})$ in c maps all occurrences of $x \in (T \cup V)$ with $\Upsilon(c, x) = i$ in $\vec{\alpha}$ to $\phi(i)$. If the result is defined (i.e., the images of adjacent variables can be concatenated), it is called an* instantiated predicate *and the result of applying ϕ to all predicates in c, if defined, is called an* instantiated clause.

We also introduce range constraint vectors, vectors of pairs of range boundary variables together with a set of constraints on these variables.

Definition 4. *Let $V_r = \{r_1, r_2, \dots\}$ be a set of range boundary variables. A* range constraint vector *of dimension k is a pair $\langle \vec{\rho}, C \rangle$ where a) $\vec{\rho} \in (V_r^2)^k$; we define $V_r(\vec{\rho})$ as the set of range boundary variables occurring in $\vec{\rho}$. b) C is a set of constraints c_r that have one of the following forms: $r_1 = r_2$, $k = r_1$, $r_1 + k = r_2$, $k \leq r_1$, $r_1 \leq k$, $r_1 \leq r_2$ or $r_1 + k \leq r_2$ for $r_1, r_2 \in V_r(\vec{\rho})$ and $k \in \mathbb{N}$.*

We say that a range vector ϕ satisfies a range constraint vector $\langle \rho, C \rangle$ iff ϕ and ρ are of the same dimension k and there is a function $f : V_r \to \mathbb{N}$ that maps $\rho(i).l$ to $\phi(i).l$ and $\rho(i).r$ to $\phi(i).r$ for all $1 \leq i \leq k$ such that all constraints in C are satisfied. Furthermore, we say that a range constraint vector $\langle \rho, C \rangle$ is satisfiable iff there exists a range vector ϕ that satisfies it.

Definition 5. *For every clause c, we define its* range constraint vector $\langle \rho, C \rangle$ *w.r.t. a w with $|w| = n$ as follows: a) ρ has dimension $\Upsilon(c)$ and all range boundary variables in ρ are pairwise different. b) For all $\langle r_1, r_2 \rangle \in \rho$: $0 \leq r_1$, $r_1 \leq r_2$, $r_2 \leq n \in C$. For all occurrences x of terminals in c with $i = \Upsilon(c, x)$: $\rho(i).l + 1 = \rho(i).r \in C$. For all x, y that are variables or occurrences of terminals in c such that xy is a substring of one of the arguments in c: $\rho(\Upsilon(c, x)).r = \rho(\Upsilon(c, y)).l \in C$. These are all constraints in C.*

The range constraint vector of a clause c captures all information about boundaries forming a range, ranges containing only a single terminal, and adjacent variables/terminal occurrences in c.

An RCG derivation consists of rewriting instantiated predicates applying instantiated clauses, i.e. in every derivation step $\Gamma_1 \Rightarrow_w \Gamma_2$, we replace the lefthand side of an instantiated clause with its righthand side (w.r.t. a word w). The *language* of an RCG G is the set of strings that can be reduced to the empty word: $L(G) = \{w \mid S(\langle 0, |w| \rangle) \stackrel{+}{\Rightarrow}_{G,w} \varepsilon\}$.

The expressive power of RCG lies beyond mild context-sensitivity. As an example, consider the RCG from Fig. 3 that generates a language that is not semilinear.

For simplicity, we assume in the following without loss of generality that empty arguments (ε) occur only in clauses whose righthand sides are empty.[2]

3 Directional Bottom-Up Chart Parsing

In our directional CYK algorithm, we move a dot through the righthand side of a clause. We therefore have *passive* items $[A, \phi]$ where A is a predicate and ϕ a range vector of dimension $dim(A)$ and *active* items. In the latter, while traversing the righthand side of the clause, we keep a record of the left and right boundaries already found for variables and terminal occurrences. This is achieved by subsequently enriching the range constraint vector of the clause. Active items have the form $[A(\vec{x}) \to \Phi \bullet \Psi, \langle \rho, C \rangle]$ with $A(\vec{x}) \to \Phi\Psi$ a clause, $\Phi\Psi \neq \varepsilon$, $\Upsilon(A(\vec{x} \to \Phi\Psi)) = j$ and $\langle \rho, C \rangle$ a range constraint vector of dimension j. We require that $\langle \rho, C \rangle$ be satisfiable.[3]

[2] Any RCG can be easily transformed into an RCG satisfying this condition: Introduce a new unary predicate Eps with a clause $Eps(\varepsilon) \to \varepsilon$. Then, for every clause c with righthand side not ε, replace every argument ε that occurs in c with a new variable X (each time a distinct one) and add the predicate $Eps(X)$ to the righthand side of c.

[3] Items that are distinguished from each other only by a bijection of the range variables are considered equivalent. I.e., if the application of a rule yields a new item such that an equivalent one has already been generated, this new one is not added to the set of partial results.

Scan: $\dfrac{}{[A,\phi]}$ $A(\vec{x}) \to \varepsilon \in P$ with instantiation ψ such that $\psi(A(\vec{x})) = A(\phi)$

Initialize: $\dfrac{}{[A(\vec{x}) \to \bullet\Phi, \langle\rho, C\rangle]}$ $A(\vec{x}) \to \Phi \in P$ with range constraint vector $\langle\rho, C\rangle, \Phi \neq \varepsilon$

Complete:

$$\frac{[B,\phi_B],\ [A(\vec{x}) \to \Phi \bullet B(x_1...y_1,...,x_k...y_k)\Psi, \langle\rho, C\rangle]}{[A(\vec{x}) \to \Phi B(x_1...y_1,...,x_k...y_k) \bullet \Psi, \langle\rho, C'\rangle]}$$

where $C' = C \cup \{\phi_B(j).l = \rho(\Upsilon(x_j)).l,\ \phi_B(j).r = \rho(\Upsilon(y_j)).r \mid 1 \leq j \leq k\}$.

Convert: $\dfrac{[A(\vec{x}) \to \Psi\bullet, \langle\rho, C\rangle]}{[A,\phi]}$ $A(\vec{x}) \to \Psi \in P$ with an instantiation ψ that satisfies $\langle\rho, C\rangle$, $\psi(A(\vec{x})) = A(\phi)$

Goal: $[S, (\langle 0, n\rangle)]$

Figure 1: CYK deduction rules

The deduction rules are shown in Fig. 1. The first rule scans the yields of terminating clauses. **Initialize** introduces clauses with the dot on the left of the righthand side. **Complete** moves the dot over a predicate provided a corresponding passive item has been found. **Convert** turns an active item with the dot at the end into a passive item.

4 The Earley Algorithm

We now add top-down prediction to our algorithm. Active items are as above. Passive items have an additional flag p or c depending on whether the item is predicted or completed, i.e., they either have the form $[A, \langle\rho, C\rangle, p]$ where $\langle\rho, C\rangle$ is a range constraint vector of dimension $dim(A)$, or the form $[A, \phi, c]$ where ϕ is a range vector of dimension $dim(A)$.

Initialize: $\dfrac{}{[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, n = r_2\}\rangle, p]}$

Predict-rule:

$$\frac{[A, \langle\rho, C\rangle, p]}{[A(x_1 \dots y_1, \dots, x_k \dots y_k) \to \bullet\Psi, \langle\rho', C'\rangle]}$$

where $\langle\rho', C'\rangle$ is obtained from the range constraint vector of the clause $A(x_1 \dots y_1, \dots, x_k \dots y_k) \to \Psi$ by taking all constraints from C, mapping all $\rho(i).l$ to $\rho'(\Upsilon(x_i)).l$ and all $\rho(i).r$ to $\rho'(\Upsilon(y_i)).r$, and then adding the resulting constraints to the range constraint vector of the clause.

Predict-pred:

$$\frac{[A(...) \to \Phi \bullet B(x_1...y_1,...,x_k...y_k)\Psi, \langle\rho, C\rangle]}{[B, \langle\rho', C'\rangle, p]}$$

where $\rho'(i).l = \rho(\Upsilon(x_i)).l$, $\rho'(i).r = \rho(\Upsilon(y_i)).r$ for all $1 \leq i \leq k$ and $C' = \{c \mid c \in C, c$ contains only range variables from $\rho'\}$.

Scan: $\dfrac{[A, \langle\rho, C\rangle, p]}{[A, \phi, c]}$ $A(\vec{x}) \to \varepsilon \in P$ with an instantiation ψ satisfying $\langle\rho, C\rangle$ such that $\psi(A(\vec{x})) = A(\phi)$

Figure 2: Earley deduction rules

The deduction rules are listed in Fig. 2. The axiom is the prediction of an S ranging over the entire input (**initialize**). We have two predict operations: **Predict-rule** predicts active items with the dot on the left of the righthand side, for a given predicted passive item. **Predict-pred** predicts a passive item for the predicate following the dot in an active item. **Scan** is applied whenever a predicted predicate can be derived by an ε-clause. The rules **complete** and **convert** are the ones from the CYK algorithm except that we add flags c to the passive items occurring in these rules. The **goal** is again $[S, (\langle 0, n\rangle), c]$.

To understand how this algorithm works, consider the example in Fig. 3. The crucial property of this algorithm, in contrast to previous approaches, is the dynamic updating of a set of constraints on range boundaries. We can leave range boundaries unspecified and compute their values in a more incremental fashion instead of guessing all ranges of a clause at once at prediction.[4]

For evaluation, we have implemented a directional top-down algorithm where range boundaries are guessed at prediction (this is essentially the algorithm described in Boullier (2000)), and the new Earley-style algorithm. The algorithms were tested on different words of the language $L = \{a^{2^n} | n \leq 0\}$. Table 1 shows the number of generated items.

Word	Earley	TD	Word	Earley	TD
a^2	15	21	a^{16}	100	539
a^4	30	55	a^{30}	155	1666
a^8	55	164	a^{32}	185	1894
a^9	59	199	a^{64}	350	6969

Table 1: Items generated by both algorithms

Clearly, range boundary constraint propagation increases the amount of information transported in single items and thereby decreases considerably the number of generated items.

5 Conclusion and future work

We have presented a new CYK and Earley parsing algorithms for the full class of RCG. The crucial difference between previously proposed top-down RCG parsers and the new Earley-style algorithm is that while the former compute all clause instantiations during **predict** operations, the latter

[4] Of course, the use of constraints makes comparisons between items more complex and more expensive which means that for an efficient implementation, an integer-based representation of the constraints and adequate techniques for constraint solving are required.

Grammar for $\{a^{2^n} \mid n > 0\}$: $S(XY) \rightarrow S(X)eq(X,Y)$, $S(a_1) \rightarrow \varepsilon$, $eq(a_1X, a_2Y) \rightarrow eq(X,Y)$, $eq(a_1, a_2) \rightarrow \varepsilon$
Parsing trace for $w = aa$:

	Item	Rule
1	$[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, r_1 \leq r_2, 2 = r_2\}\rangle, p]$	initialize
2	$[S(XY) \rightarrow \bullet S(X)eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, 2 = Y.r\}]$	predict-rule from 1
3	$[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, r_1 \leq r_2\}\rangle, p]$	predict-pred from 2
4	$[S, (\langle 0, 1\rangle), c]$	scan from 3
5	$[S(XY) \rightarrow \bullet S(X)eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, \}]$	predict-rule from 3
6	$[S(XY) \rightarrow S(X) \bullet eq(X,Y), \{\ldots, 0 = X.l, 2 = Y.r, 1 = X.r\}]$	complete 2 with 4
7	$[S(XY) \rightarrow S(X) \bullet eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, 1 = X.r\}]$	complete 5 with 4
8	$[eq, \langle(\langle r_1, r_2\rangle, \langle r_3, r_4\rangle), \{r_1 \leq r_2, r_2 = r_3, r_3 \leq r_4, 0 = r_1, 2 = r_4, 1 = r_2\}\rangle]$	predict-pred from 6
9	$[eq(a_1X, a_2Y) \rightarrow \bullet eq(X,Y), \{a_1.l + 1 = a_1.r, a_1.r = X.l, X.l \leq X.r,$ $a_2.l + 1 = a_2.r, a_2.r = Y.l, Y.l \leq Y.r, X.r = a_2.l, 0 = a_1.l, 1 = X.r, 2 = Y.r\}]$	predict-rule from 8
	...	
10	$[eq, (\langle 0, 1\rangle, \langle 1, 2\rangle), c]$	scan 8
11	$[S(XY) \rightarrow S(X)eq(X,Y) \bullet, \{\ldots, 0 = X.l, 2 = Y.r, 1 = X.r, 1 = Y.l\}]$	complete 6 with 10
12	$[S, (\langle 0, 2\rangle), c]$	convert 11

Figure 3: Trace of a sample Earley parse

avoids this using a technique of dynamic updating of a set of constraints on range boundaries. Experiments show that this significantly decreases the number of generated items, which confirms that range boundary constraint propagation is a viable method for a lazy computation of ranges.

The Earley parser could be improved by allowing to process the predicates of the righthand sides of clauses in any order, not necessarily from left to right. This way, one could process predicates whose range boundaries are better known first. We plan to include this strategy in future work.

References

François Barthélemy, Pierre Boullier, Philippe Deschamp, and Éric de la Clergerie. 2001. Guided parsing of Range Concatenation Languages. In *Proceedings of ACL*, pages 42–49.

Tilman Becker, Aravind K. Joshi, and Owen Rambow. 1991. Long-distance scrambling and tree adjoining grammars. In *Proceedings of EACL*.

Tilman Becker, Owen Rambow, and Michael Niv. 1992. The Derivationel Generative Power of Formal Systems or Scrambling is Beyond LCFRS. Technical Report IRCS-92-38, Institute for Research in Cognitive Science, University of Pennsylvania.

E. Bertsch and M.-J. Nederhof. 2001. On the complexity of some extensions of RCG parsing. In *Proceedings of IWPT 2001*, pages 66–77, Beijing, China.

Pierre Boullier. 1999. Chinese numbers, mix, scrambling, and range concatenation grammars. In *Proceedings of EACL*, pages 53–60, Bergen, Norway.

Pierre Boullier. 2000. Range concatenation grammars. In *Proceedings of IWPT 2000*, pages 53–64, Trento.

Håkan Burden and Peter Ljunglöf. 2005. Parsing linear context-free rewriting systems. In *Proceedings of IWPT 2005*, pages 11–17, Vancouver.

Laura Kallmeyer, Timm Lichte, Wolfgang Maier, Yannick Parmentier, and Johannes Dellert. 2008. Developing an MCTAG for German with an RCG-based parser. In *Proceedings of LREC-2008*, Marrakech, Morocco.

Wolfgang Maier and Anders Søgaard. 2008. Treebanks and mild context-sensitivity. In *Proceedings of the 13th Conference on Formal Grammar 2008*, Hamburg, Germany.

Jens Michaelis and Marcus Kracht. 1996. Semilinearity as a Syntactic Invariant. In *Logical Aspects of Computational Linguistics*, Nancy.

Daniel Radzinski. 1991. Chinese number-names, tree adjoining languages, and mild context-sensitivity. *Computational Linguistics*, 17:277–299.

Benoît Sagot. 2005. Linguistic facts as predicates over ranges of the sentence. In *Proceedings of LACL 05*, number 3492 in Lecture Notes in Computer Science, pages 271–286, Bordeaux, France. Springer.

Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1& 2):3–36.

Anders Søgaard. 2008. Range concatenation grammars for translation. In *Proceedings of COLING*, Manchester, England.

K. Vijay-Shanker, David Weir, and Aravind Joshi. 1987. Characterising structural descriptions used by various formalisms. In *Proceedings of ACL*.

Eric Villemonte de la Clergerie. 2002. Parsing mildly context-sensitive languages with thread automata. In *Proceedings of COLING*, Taipei, Taiwan.

Using Syntax to Disambiguate Explicit Discourse Connectives in Text*

Emily Pitler and Ani Nenkova
Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
epitler,nenkova@seas.upenn.edu

Abstract

Discourse connectives are words or phrases such as *once*, *since*, and *on the contrary* that explicitly signal the presence of a discourse relation. There are two types of ambiguity that need to be resolved during discourse processing. First, a word can be ambiguous between discourse or non-discourse usage. For example, *once* can be either a temporal discourse connective or a simply a word meaning "formerly". Secondly, some connectives are ambiguous in terms of the relation they mark. For example *since* can serve as either a temporal or causal connective. We demonstrate that syntactic features improve performance in both disambiguation tasks. We report state-of-the-art results for identifying discourse vs. non-discourse usage and human-level performance on sense disambiguation.

1 Introduction

Discourse connectives are often used to explicitly mark the presence of a discourse relation between two textual units. Some connectives are largely unambiguous, such as *although* and *additionally*, which are almost always used as discourse connectives and the relations they signal are unambiguously identified as comparison and expansion, respectively. However, not all words and phrases that can serve as discourse connectives have these desirable properties.

Some linguistic expressions are ambiguous between DISCOURSE AND NON-DISCOURSE USAGE. Consider for example the following sentences containing *and* and *once*.

(1a) Selling picked up as previous buyers bailed out of their positions *and* aggressive short sellers– anticipating further declines–moved in.

(1b) My favorite colors are blue *and* green.

(2a) The asbestos fiber, crocidolite, is unusually resilient *once* it enters the lungs, with even brief exposures to it causing symptoms that show up decades later, researchers said.

(2b) A form of asbestos *once* used to make Kent cigarette filters has caused a high percentage of cancer deaths among a group of workers exposed to it more than 30 years ago, researchers reported.

In sentence (1a), *and* is a discourse connective between the two clauses linked by an elaboration/expansion relation; in sentence (1b), the occurrence of *and* is non-discourse. Similarly in sentence (2a), *once* is a discourse connective marking the temporal relation between the clauses "The asbestos fiber, crocidolite is unusually resilient" and "it enters the lungs". In contrast, in sentence (2b), *once* occurs with a non-discourse sense, meaning "formerly" and modifying "used".

The only comprehensive study of discourse vs. non-discourse usage in written text[1] was done in the context of developing a complete discourse parser for unrestricted text using surface features (Marcu, 2000). Based on the findings from a corpus study, Marcu's parser "ignored both cue phrases that had a sentential role in a majority of the instances in the corpus and those that were too ambiguous to be explored in the context of a surface-based approach".

The other ambiguity that arises during discourse processing involves DISCOURSE RELATION SENSE. The discourse connective *since* for

This work was partially supported by NSF grants IIS-0803159, IIS-0705671 and IGERT 0504487.

[1]The discourse vs. non-discourse usage ambiguity is even more problematic in spoken dialogues because there the number of potential discourse markers is greater than that in written text, including common words such as *now*, *well* and *okay*. Prosodic and acoustic features are the most powerful indicators of discourse vs. non-discourse usage in that genre (Hirschberg and Litman, 1993; Gravano et al., 2007)

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 13–16,
Suntec, Singapore, 4 August 2009.

instance can signal either a temporal or a causal relation as shown in the following examples from Miltsakaki et al. (2005):

(3a) There have been more than 100 mergers and acquisitions within the European paper industry *since* the most recent wave of friendly takeovers was completed in the U.S. in 1986.

(3b) It was a far safer deal for lenders *since* NWA had a healthier cash flow and more collateral on hand.

Most prior work on relation sense identification reports results obtained on data consisting of both explicit and implicit relations (Wellner et al., 2006; Soricut and Marcu, 2003). Implicit relations are those inferred by the reader in the absence of a discourse connective and so are hard to identify automatically. Explicit relations are much easier (Pitler et al., 2008).

In this paper, we explore the predictive power of syntactic features for both the discourse vs. non-discourse usage (Section 3) and discourse relation sense (Section 4) prediction tasks for explicit connectives in written text. For both tasks we report high classification accuracies close to 95%.

2 Corpus and features

2.1 Penn Discourse Treebank

In our work we use the Penn Discourse Treebank (PDTB) (Prasad et al., 2008), the largest public resource containing discourse annotations. The corpus contains annotations of 18,459 instances of 100 explicit discourse connectives. Each discourse connective is assigned a sense from a three-level hierarchy of senses. In our experiments we consider only the top level categories: Expansion (one clause is elaborating information in the other), Comparison (information in the two clauses is compared or contrasted), Contingency (one clause expresses the cause of the other), and Temporal (information in two clauses are related because of their timing). These top-level discourse relation senses are general enough to be annotated with high inter-annotator agreement and are common to most theories of discourse.

2.2 Syntactic features

Syntactic features have been extensively used for tasks such as argument identification: dividing sentences into elementary discourse units among which discourse relations hold (Soricut and Marcu, 2003; Wellner and Pustejovsky, 2007; Fisher and Roark, 2007; Elwell and Baldridge, 2008). Syntax has not been used for discourse vs. non-discourse disambiguation, but it is clear from the examples above that discourse connectives appear in specific syntactic contexts.

The syntactic features we used were extracted from the gold standard Penn Treebank (Marcus et al., 1994) parses of the PDTB articles:

Self Category The highest node in the tree which dominates the words in the connective but nothing else. For single word connectives, this might correspond to the POS tag of the word, however for multi-word connectives it will not. For example, the cue phrase *in addition* is parsed as (PP (IN In) (NP (NN addition))). While the POS tags of "in" and "addition" are preposition and noun, respectively, together the Self Category of the phrase is prepositional phrase.

Parent Category The category of the immediate parent of the Self Category. This feature is especially helpful for disambiguating cases similar to example (1b) above in which the parent of *and* would be an NP (the noun phrase "blue and green"), which will rarely be the case when *and* has a discourse function.

Left Sibling Category The syntactic category of the sibling immediately to the left of the Self Category. If the left sibling does not exist, this features takes the value "NONE". Note that having no left sibling implies that the connective is the first substring inside its Parent Category. In example (1a), this feature would be "NONE", while in example (1b), the left sibling of *and* is "NP".

Right Sibling Category The syntactic category of the sibling immediately to the right of the Self Category. English is a right-branching language, and so dependents tend to occur after their heads. Thus, the right sibling is particularly important as it is often the dependent of the potential discourse connective under investigation. If the connective string has a discourse function, then this dependent will often be a clause (SBAR). For example, the discourse usage in "*After* I went to the store, I went home" can be distinguished from the non-discourse usage in "*After* May, I will go on vacation" based on the categories of their right siblings.

Just knowing the syntactic category of the right sibling is sometimes not enough; experiments on the development set showed improvements by including more features about the right sibling.

Consider the example below:

(4) NASA won't attempt a rescue; instead, it will try to predict whether any of the rubble will smash to the ground

and where.

The syntactic category of "where" is SBAR, so the set of features above could not distinguish the single word "where" from a full embedded clause like "I went to the store". In order to address this deficiency, we include two additional features about the contents of the right sibling, **Right Sibling Contains a VP** and **Right Sibling Contains a Trace**.

3 Discourse vs. non-discourse usage

Of the 100 connectives annotated in the PDTB, only 11 appear as a discourse connective more than 90% of the time: *although*, *in turn*, *afterward*, *consequently*, *additionally*, *alternatively*, *whereas*, *on the contrary*, *if and when*, *lest*, and *on the one hand...on the other hand*. There is quite a range among the most frequent connectives: *although* appears as a discourse connective 91.4% of the time, while *or* only serves a discourse function 2.8% of the times it appears.

For training and testing, we used explicit discourse connectives annotated in the PDTB as positive examples and occurrences of the same strings in the PDTB texts that were not annotated as explicit connectives as negative examples.

Sections 0 and 1 of the PDTB were used for development of the features described in the previous section. Here we report results using a maximum entropy classifier[2] using ten-fold cross-validation over sections 2-22.

The results are shown in Table 3. Using the string of the connective as the only feature sets a reasonably high baseline, with an f-score of 75.33% and an accuracy of 85.86%. Interestingly, using only the syntactic features, ignoring the identity of the connective, is even better, resulting in an f-score of 88.19% and accuracy of 92.25%. Using both the connective and syntactic features is better than either individually, with an f-score of 92.28% and accuracy of 95.04%.

We also experimented with combinations of features. It is possible that different connectives have different syntactic contexts for discourse usage. Including pair-wise interaction features between the connective and each syntactic feature (features like *connective=also-RightSibling=SBAR*) raised the f-score about 1.5%, to 93.63%. Adding interaction terms between pairs of syntactic features raises the f-score slightly more, to 94.19%. These results amount to a 10% absolute improvement over those obtained by Marcu (2000) in his corpus-based approach which achieves an f-score of 84.9%[3] for identifying discourse connectives in text. While bearing in mind that the evaluations were done on different corpora and so are not directly comparable, as well as that our results would likely drop slightly if an automatic parser was used instead of the gold-standard parses, syntactic features prove highly beneficial for discourse vs. non-discourse usage prediction, as expected.

Features	**Accuracy**	**f-score**
(1) Connective Only	85.86	75.33
(2) Syntax Only	92.25	88.19
(3) Connective+Syntax	95.04	92.28
(3)+Conn-Syn Interaction	95.99	93.63
(3)+Conn-Syn+Syn-Syn Interaction	**96.26**	**94.19**

Table 1: Discourse versus Non-discourse Usage

4 Sense classification

While most connectives almost always occur with just one of the senses (for example, *because* is almost always a Contingency), a few are quite ambiguous. For example *since* is often a Temporal relation, but also often indicates Contingency.

After developing syntactic features for the discourse versus non-discourse usage task, we investigated whether these same features would be useful for sense disambiguation.

Experiments and results We do classification between the four senses for each explicit relation and report results on ten-fold cross-validation over sections 2-22 of the PDTB using a Naive Bayes classifier[4].

Annotators were allowed to provide two senses for a given connective; in these cases, we consider either sense to be correct[5]. Contingency and Temporal are the senses most often annotated together. The connectives most often doubly annotated in the PDTB are *when* (205/989), *and* (183/2999), and *as* (180/743).

Results are shown in Table 4. The sense classification accuracy using just the connective is already quite high, 93.67%. Incorporating the syntactic features raises performance to 94.15% accu-

[2]http://mallet.cs.umass.edu

[3]From the reported precision of 89.5% and recall of 80.8%

[4]We also ran a MaxEnt classifier and achieved quite similar but slightly lower results.

[5]Counting only the first sense as correct leads to about 1% lower accuracy.

Features	Accuracy
Connective Only	93.67
Connective+Syntax+Conn-Syn	**94.15**
Interannotator agreement on sense class (Prasad et al., 2008)	*94*

Table 2: Four-way sense classification of explicits

racy. While the improvement is not huge, note that we seem to be approaching a performance ceiling. The human inter-annotator agreement on the top level sense class was also 94%, suggesting further improvements may not be possible. We provide some examples to give a sense of the type of errors that still occur.

Error Analysis While Temporal relations are the least frequent of the four senses, making up only 19% of the explicit relations, more than half of the errors involve the Temporal class. By far the most commonly confused pairing was Contingency relations being classified as Temporal relations, making up 29% of our errors.

A random example of each of the most common types of errors is given below.

(5) *Builders get away with using sand and financiers junk* [when] **society decides it's okay, necessary even, to look the other way.** Predicted: Temporal Correct: Contingency

(6) *You get a rain at the wrong time* [and] **the crop is ruined.** Predicted: Expansion Correct: Contingency

(7) *In the nine months, imports rose 20% to 155.039 trillion lire* [and] **exports grew 18% to 140.106 trillion lire.** Predicted: Expansion Correct: Comparison

(8) *[The biotechnology concern said] Spanish authorities must still clear the price for the treatment* [but] **that it expects to receive such approval by year end.** Predicted: Comparison Correct: Expansion

Examples (6) and (7) show the relatively rare scenario when *and* does not signal expansion, and Example (8) shows *but* indicating a sense besides comparison. In these cases where the connective itself is not helpful in classifying the sense of the relation, it may be useful to incorporate features that were developed for classifying implicit relations (Sporleder and Lascarides, 2008).

5 Conclusion

We have shown that using a few syntactic features leads to state-of-the-art accuracy for discourse vs. non-discourse usage classification. Including syntactic features also helps sense class identification, and we have already attained results at the level of human annotator agreement. These results taken together show that explicit discourse connectives can be identified automatically with high accuracy.

References

R. Elwell and J. Baldridge. 2008. Discourse connective argument identification with connective specific rankers. In *Proceedings of the International Conference on Semantic Computing, Santa Clara, CA*.

S. Fisher and B. Roark. 2007. The utility of parse-derived features for automatic discourse segmentation. In *Proceedings of ACL*, pages 488–495.

A. Gravano, S. Benus, H. Chavez, J. Hirschberg, and L. Wilcox. 2007. On the role of context and prosody in the interpretation of 'okay'. In *Proceedings of ACL*, pages 800–807.

J. Hirschberg and D. Litman. 1993. Empirical studies on the disambiguation of cue phrases. *Computational linguistics*, 19(3):501–530.

D. Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

E. Miltsakaki, N. Dinesh, R. Prasad, A. Joshi, and B. Webber. 2005. Experiments on sense annotation and sense disambiguation of discourse connectives. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*.

E. Pitler, M. Raghupathy, H. Mehta, A. Nenkova, A. Lee, and A. Joshi. 2008. Easily identifiable discourse relations. In *COLING, short paper*.

R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC'08*.

R. Soricut and D. Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *HLT-NAACL*.

C. Sporleder and A. Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14:369–416.

B. Wellner and J. Pustejovsky. 2007. Automatically identifying the arguments of discourse connectives. In *Proceedings of EMNLP-CoNLL*, pages 92–101.

B. Wellner, J. Pustejovsky, C. Havasi, A. Rumshisky, and R. Sauri. 2006. Classification of discourse coherence relations: An exploratory study using multiple knowledge sources. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*.

Hybrid Approach to User Intention Modeling for Dialog Simulation

Sangkeun Jung, Cheongjae Lee, Kyungduk Kim, Gary Geunbae Lee
Department of Computer Science and Engineering
Pohang University of Science and Technology(POSTECH)
{hugman, lcj80, getta, gblee}@postech.ac.kr

Abstract

This paper proposes a novel user intention simulation method which is a data-driven approach but able to integrate diverse user discourse knowledge together to simulate various type of users. In Markov logic framework, logistic regression based data-driven user intention modeling is introduced, and human dialog knowledge are designed into two layers such as domain and discourse knowledge, then it is integrated with the data-driven model in generation time. Cooperative, corrective and self-directing discourse knowledge are designed and integrated to mimic such type of users. Experiments were carried out to investigate the patterns of simulated users, and it turned out that our approach was successful to generate user intention patterns which are not only unseen in the training corpus and but also personalized in the designed direction.

1 Introduction

User simulation techniques are widely used for learning optimal dialog strategies in a statistical dialog management framework and for automated evaluation of spoken dialog systems. User simulation can be layered into the user intention level and user surface (utterance) level. This paper proposes a novel intention level user simulation technique.

In recent years, a data-driven user intention modeling is widely used since it is domain- and language independent. However, the problem of data-driven user intention simulation is the limitation of user patterns. Usually, the response patterns from data-driven simulated user tend to be limited to the training data. Therefore, it is not easy to simulate unseen user intention patterns, which is quite important to evaluate or learn optimal dialog policies. Another problem is poor user type controllability in a data-driven method. Sometimes, developers need to switch testers between various type of users such as cooperative, uncooperative or novice user and so on to expose their dialog system to various users.

For this, we introduce a novel data-driven user intention simulation method which is powered by human dialog knowledge in Markov logic formulation (Richardson and Domingos, 2006) to add diversity and controllability to data-driven intention simulation.

2 Related work

Data-driven intention modeling approach uses statistical methods to generate the user intention given discourse information (history). The advantage of this approach lies in its simplicity and in that it is domain- and language independency. *N*-gram based approaches (Eckert et al., 1997, Levin et al., 2000) and other approaches (Scheffler and Young, 2001, Pietquin and Dutoit, 2006, Schatzmann et al., 2007) are introduced. There has been some work on combining rules with statistical models especially for system side dialog management (Heeman, 2007, Henderson et al., 2008). However, little prior research has tried to use both knowledge and data-driven methods together in a single framework especially for user intention simulation.

In this research, we introduce a novel data-driven user intention modeling technique which can be diversified or personalized by integrating human discourse knowledge which is represented in first-order logic in a single framework. In the framework, diverse type of user knowledge can be easily designed and selectively integrated into data-driven user intention simulation.

3 Overall architecture

The overall architecture of our user simulator is shown in Fig. 1. The user intention simulator accepts the discourse circumstances with system intention as input and generates the next user intention. The user utterance simulator constructs a corresponding user sentence to express the given user intention. The simulated user sentence is fed to the automatic speech recognition (ASR) channel simulator, which then adds noises to the utterance. The noisy utterance is passed to a dialog system which consists of spoken language understanding (SLU) and dialog management (DM) modules. In this research, the user utterance simulator and ASR channel simulator are developed using the method of (Jung et al., 2009).

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 17–20,
Suntec, Singapore, 4 August 2009.

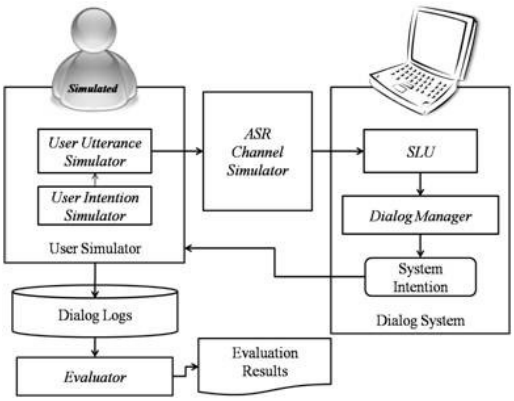


Fig. 1 Overall architecture of dialog simulation

4 Markov logic

Markov logic is a probabilistic extension of finite first-order logic (Richardson and Domingos, 2006). A *Markov Logic Network* (MLN) combines first-order logic and probabilistic graphical models in a single representation.

An MLN can be viewed as a *template* for constructing Markov networks. From the above definition, the probability distribution over possible worlds *x* specified by the ground Markov network is given by

$$P(X = x) = \frac{1}{Z}\exp(\sum_{i=1}^{F} w_i n_i(x))$$

where F is the number of formulas in the MLN and $n_i(x)$ is the number of true groundings of F_i in x. As formula weights increase, an MLN increasingly resembles a purely logical KB, becoming equivalent to one in the limit of all infinite weights. General algorithms for inference and learning in Markov logic are discussed in (Richardson and Domingos, 2006).

Since Markov logic is a first-order knowledge base with a weight attached to each formula, it provides a theoretically fine framework integrating a statistically learned model with logically designed and inducted human knowledge. So the framework can be used for building up a hybrid user modeling with the advantages of knowledge-based and data-driven models.

5 User intention modeling in Markov logic

The task of user intention simulation is to generate subsequent user intentions given current discourse circumstances. Therefore, user intention simulation can be formulated in the probabilistic form *P(userIntention | context)*.

In this research, we define the user intention state *userIntention* = [dialog_act, main_goal, component_slot], where dialog_act is a domain-independent label of an utterance at the level of illocutionary force (e.g. statement, request, wh_question) and main_goal is the domain-specific user goal of an utterance (e.g. give_something, tell_purpose). Component slots represent domain-specific named-entities in the utterance. For example, in the user intention state for the utterance "*I want to go to city hall*" (Fig. 2), the combination of each slot of semantic frame represents the user intention symbol. In this example, the state symbol is 'request+search_loc+[loc_name]'. Dialogs on car navigation deal with support for the information and selection of the desired destination.

The first-order language-based predicates which are related with discourse context information and with generating the next user intention are as follows:

- ***User intention simulation related predicates***
  `GenerateUserIntention`(context,userIntention)
- ***Discourse context related predicates***
  `hasIntention`(context, userIntention)
  `hasDialogAct`(context, dialogAct)
  `hasMainGoal`(context, mainGoal)
  `hasEntity`(context, entity)
  `isFilledComponent`(context,entity)
  `updatedEntity`(contetx, entity)
  `hasNumDBResult`(context, numDBResult)
  `hasSystemAct`(context, systemAct)
  `hasSystemActAttr`(context, sytemActAttr)
  `isSubTask`(context, subTask)

For example, after the following fragment of dialog for the car navigation domain,

User(01) : Where are Chinese restaurants?
// dialog_act=wh_question
// main_goal=search_loc
// named_entity[loc_keyword]=Chinese_restaurant
Sys(01) : There are Buchunsung and Idongbanjum in Daeidong.
// system_act=inform
// target action attribute=name,address

the discourse context which is passed to the user simulator is illustrated in Fig. 3.

Notice that the context information is composed of semantic frame (SF), discourse history (DH) and previous system intention (SI). '`isFilledComponent`' predicate indicates which component slots are filled during the discourse. '`updatedEntity`' predicate is true if the corresponding named entity is newly updated. '`hasSystemAct`' and '`hasSystemActAttr`' predicate represent previous system intention and mentioned attributes.

raw user utterance	*I want to go to city hall.*
dialog_act	*request*
main_goal	*search_loc*
component.[loc_name]	*cityhall*

Fig. 2 Semantic frame for user intention simulation on car navigation domain.

SF	`hasIntention`("ct_01", "request+search_loc+loc_name") `hasDialogAct`("ct_01","wh_question") `hasMainGoal`("ct_01", "search_loc") `hasEntity`("ct_01", "loc_keyword")
DH	`isFilledComponent`("ct_01", "loc_keyword) `!isFilledComponent`("ct_01", "loc_address) `!isFilledComponent`("ct_01", "loc_name") `!isFilledComponent`("ct_01", "route_type") `updatedEntity`("ct_01", "loc_keyword")
SI	`hasNumDBResult`("ct_01", "many") `hasSystemAct`("ct_01", "inform") `hasSystemActAttr`("ct_01", "address,name")

Fig. 3 Example of discourse context in car navigation domain. SF=Semantic Frame, DH=Discourse History, SI=System Intention.

5.1 Data-driven user intention modeling in Markov logic

The formulas are defined between the predicates which are related with discourse context information and corresponding user intention. The formulas for user intention modeling based on logistic regression are as follows:

∀ct, pui, ui `hasIntention`(ct, pui)[1]
=> `GenerateUserIntention`(ct, ui)
∀ct, da, ui `hasDialogAct`(ct, da) => `GenerateUserIntention`(ct,ui)
∀ct, mg, ui `hasMainGoal`(ct, mg) => `GenerateUserIntention`(ct,ui)
∀ct, en, ui `hasEntity`(ct, en) =>`GenerateUserIntention`(ct,ui)
∀ct, en, ui `isFilledComponent`(ct,en)
=> `GenerateUserIntention`(ct,ui)
∀ct, en, ui `updatedEntity`(ct, en) => `GenerateUserIntention`(ct,ui)
∀ct, dbr, ui `hasNumDBResult`(ct, dbr)
=> `GenerateUserIntention`(ct, ui)
∀ct, sa, ui `hasSystemAct`(ct, sa) =>`GenerateUserIntention`(ct, ui)
∀ct, attr, ui `hasSystemActAttr`(ct, attr)
=> `GenerateUserIntention`(ct, ui)

The weights of each formula are estimated from the data which contains the evidence (*context*) and corresponding user intention of next turn (*userIntention*).

5.2 User knowledge

In this research, the user knowledge, which is used for deciding user intention given discourse context, is layered into two levels: *domain knowledge* and *discourse knowledge*. Domain- specific and –dependent knowledge is described in domain knowledge. Discourse knowledge is more general and abstracted knowledge. It uses the domain knowledge as base knowledge. The subtask which is one of domain knowledge are defined as follows

- ***Subtask related predicates***
  `subTaskHasIntention`(subTask,userIntetion)
  `moveTo`(subtask, subTask)
  `isCompletedSubTask`(context, subTask)
  `isSubtask`(context,subTask)

'`isSubTask`' implies which subtask corresponds to the current context. '`subTaskHasIntention`' describes which subtask has which user intention. '`moveTo`' predicate implies the connection from subtask to subtask node.

Cooperative, corrective and self-directing discourse knowledge is represented in Markov logic to mimic following users.

- ***Cooperative User***: A user who is cooperative with a system by answering what the system asked.
- ***Corrective User***: A user who try to correct the misbehavior of system by jumping to or repeating specific subtask.
- ***Self-directing User***: A user who tries to say what he/she want to without considering system's suggestion.

Examples of discourse knowledge description for three types of user are shown in Fig. 4.

[1] ct: context, ui: user intention, pui: previous user intention, da: dialog act, mg: main goal, en: entity, dbr:DB result, sa: system action, attr: target attribute of system action

Both the formulas from data-driven model and formulas from discourse knowledge are used for constructing MLN in generation time.

In inference, the discourse context related predicates are given to MLN as true, then probabilities of predicate '`GenerateUserIntention`' over candidate user intention are calculated. One of example evidence predicates was shown in Fig. 3. All of the predicates of Fig. 3 are given to MLN as true. From the network, the probability of *P(userIntention | context)* is calculated.

Cooperative Knoweldge

// If system asks to specify an address explicitly, cooperative users would specify the address by jumping to the address setting subtask.

∀ ct, st `isSubTask`(ct, st) ^
`hasSytemAct`(ct, "specify") ^
`hasSystemActAttr`(ct, "address")
=> `moveTo`(st, "AddressSetting")

Corrective Knowledge

// If the current subtask fails, corrective users would repeat current subtask.

∀ ct, st `isSubTask`(ct, st) ^
¬ `isCompletedSubTask`(ct, st) ^
`subTaskHasIntention`(st, ui)
=> `GenerateUserIntention`(ct,ui)

Self-directing Knowledge

// Self-directing users do not make an utterance which is not relevant with the next subtask in their knowledge.

∀ ct, st `isSubTask`(ct, st) ^
¬ `moveTo`(st, nt) ^
`subTaskHasIntention`(nt, ui)
=> ¬ `GenerateUserIntention`(ct, ui)

Fig. 4 Example of cooperative, corrective and self-directing discourse knowledge.

	A	B	C	D	E	F	G	H
Statistical model (S)	O	O	O	O	O	O	O	O
Cooperative(CPR)		O			O	O		O
Corrective(COR)			O		O		O	O
Self-directing(SFD)				O		O	O	O

Fig. 5 Eight different users (A to H) according to the combination of knowledge.

6 Experiments

137 dialog examples from a real user and a dialog system in the car navigation domain were used to train the data-driven user intention simulator. The SLU and DM are built in the same way of (Jung et al., 2009). After the training, simulations collected 1000 dialog samples at each word error rate (WER) setting (WER=0 to 40%). The simulator model can be varied according to the combination of knowledge. We can generate eight different simulated users from A to H as Fig. 5.

The overall trend of simulated dialogs are examined by defining an *average score* function similar to the reward score commonly used in reinforcement learning-based dialog systems for measuring both a cost and task success. We give 20 points for the successful dialog state and penalize 1 point for each action performed by the user to penalize longer dialogs.

Fig. 6 shows that simulated user C which has corrective knowledge with statistical model show significantly different trend over the most of word error rate settings. For the cooperative user (B), the difference is not as large and not statistically significant. It can be analyzed that the cooperative user behaviors are relatively common patterns in human-machine dialog corpus. So, these behaviors can be already learned in statistical model (A).

Using more than two type of knowledge together shows interesting result. Using cooperative knowledge with corrective knowledge together (E) shows much different result than using each knowledge alone (B and C). In the case of using self-directing knowledge with cooperative knowledge (F), the average scores are partially increased against base line scores. However, using corrective knowledge with self-directing knowledge does not show different result. It can be thought that the corrective knowledge and self-directing knowledge are working as contradictory policy in deciding user intention. Three discourse knowledge combined user shows very interesting result. H shows much higher improvement over all simulated users, and the differences are significant results at $p \leq 0.001$.

To verify the proposed user simulation method can simulate the unseen events, the unseen rates of units were calculated. Fig. 7 shows the unseen unit rates of intention sequence. The unseen rate of n-gram varies according to the simulated user. Notice that simulated user C, E and H generates higher unseen n-gram patterns over all word error settings. These users commonly have corrective knowledge, and the patterns seem to not be present in the corpus. But the unseen patterns do not mean poor intention simulation. Higher task completion rate of C, E and H imply that these users actually generate corrective user response to make a successful conversation.

7 Conclusion

This paper presented a novel user intention simulation method which is a data-driven approach but able to integrate diverse user discourse knowledge together to simulate various type of user. A logistic regression model is used for the statistical user intention model in Markov logic. Human dialog knowledge is separated into domain and discourse knowledge, and cooperative, corrective and self-directing discourse knowledge are designed to mimic such type user. The experiment results show that the proposed user intention simulation framework actually generates natural and diverse user intention patterns what the developer intended.

Acknowledgments

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute for Information Technology Advancement) (IITA-2009-C1090-0902-0045).

model \ WER(%)	0	10	20	30	40
A:S (base line)	14.22 (0.00)	9.13 (0.00)	5.55 (0.00)	1.33 (0.00)	-1.16 (0.00)
B:S+CPR	14.39 (0.17)	9.78 (0.65)	5.38 (-0.17)	2.32† (0.99)	-1.00 (0.16)
C:S+COR	14.61† (0.40)	**10.91♠ (1.78)**	**7.28♠ (1.74)**	**2.62‡ (1.30)**	-0.81 (0.35)
D:S+SFD	**15.70♠ (1.48)**	10.10‡ (0.97)	5.51 (-0.04)	1.89 (0.56)	-0.96♠ (0.20)
E:S+CPR+COR	14.75‡ (0.53)	**10.93♠ (1.79)**	**6.88‡ (1.33)**	**2.94♠ (1.61)**	-1.06† (0.11)
F:S+CPR+SFD	**15.75♠ (1.54)**	**10.16‡ (1.02)**	5.80 (0.26)	1.88 (0.56)	**-0.03‡ (1.13)**
G:S+COR+SFD	14.39 (0.17)	9.18 (0.05)	5.04 (-0.50)	1.63 (0.31)	-1.52 (-0.36)
H:S+CPR+COR+SFD	**15.70♠ (1.48)**	**12.19♠ (3.05)**	**9.20♠ (3.65)**	**5.12♠ (3.80)**	**1.32♠ (2.48)**

Fig. 6 Average scores of user intention models over used discourse knowledge. The relative improvements against statistical models are described between parentheses. Bold cells indicate the improvements are higher than 1.0.
† : significantly different from the base line, p = 0.05,
‡ : significantly different from the base line, p = 0.01,
♠: significantly different from the base line, $p \leq 0.001$

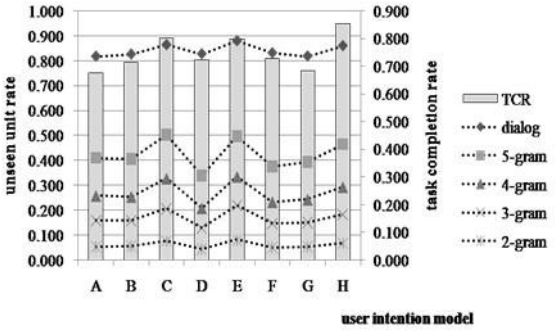


Fig. 7 Unseen user intention sequence rate and task completion rate over simulated users at word error rate of 10.

References

Eckert, W., Levin, E. and Pieraccini, R. 1997. User modeling for spoken dialogue system evaluation. *Automatic Speech Recognition and Understanding*:80-87.

Heeman, P. 2007. Combining reinforcement learning with information-state update rules. *NAACL*.

Henderson, J., Lemon, O. and Georgila, K. 2008. Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. *Comput. Linguist.,* 34(4):487-511.

Jung, S., Lee, C., Kim, K. and Lee, G.G. 2009. Data-driven user simulation for automated evaluation of spoken dialog systems. *Computer Speech & Language*.doi:10.1016/j.csl.2009.03.002.

Levin, E., Pieraccini, R. and Eckert, W. 2000. A stochastic model of human-machine interaction for learning dialog-strategies. *IEEE Transactions on Speech and Audio Processing,* 8(1):11-23.

Pietquin, O. and Dutoit, T. 2006. A Probabilistic Framework for Dialog Simulation and Optimal Strategy Learning. *IEEE Transactions on Audio, Speech and Language Processing,* 14(2):589-599.

Richardson, M. and Domingos, P. 2006. Markov logic networks. *Machine Learning,* 62(1):107-136.

Schatzmann, J., Thomson, B. and Young, S. 2007. Statistical User Simulation with a Hidden Agenda. *SIGDial*.

Scheffler, K. and Young, S. 2001. Corpus-based dialogue simulation for automatic strategy learning and evaluation. *NAACL Workshop on Adaptation in Dialogue Systems*:64-70.

Homophones and Tonal Patterns in English-Chinese Transliteration

Oi Yee Kwong
Department of Chinese, Translation and Linguistics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong
Olivia.Kwong@cityu.edu.hk

Abstract

The abundance of homophones in Chinese significantly increases the number of similarly acceptable candidates in English-to-Chinese transliteration (*E2C*). The dialectal factor also leads to different transliteration practice. We compare *E2C* between Mandarin Chinese and Cantonese, and report work in progress for dealing with homophones and tonal patterns despite potential skewed distributions of individual Chinese characters in the training data.

1 Introduction

This paper addresses the problem of automatic English-Chinese forward transliteration (referred to as *E2C* hereafter).

There are only a few hundred Chinese characters commonly used in names, but their combination is relatively free. Such flexibility, however, is not entirely ungoverned. For instance, while the Brazilian striker Ronaldo is rendered as 朗拿度 *long5-naa4-dou6* in Cantonese, other phonetically similar candidates like 朗娜度 *long5-naa4-dou6* or 郎拿刀 *long4-naa4-dou1*[1] are least likely. Beyond linguistic and phonetic properties, many other social and cognitive factors such as dialect, gender, domain, meaning, and perception, are simultaneously influencing the naming process and superimposing on the surface graphemic correspondence.

The abundance of homophones in Chinese further complicates the problem. Past studies on phoneme-based *E2C* have reported their adverse effects (e.g. Virga and Khudanpur, 2003). Direct orthographic mapping (e.g. Li *et al.*, 2004), making use of individual Chinese graphemes, tends to overcome the problem and model the character choice directly. Meanwhile, Chinese is a typical tonal language and the tone information can help distinguish certain homophones. Phoneme mapping studies seldom make use of tone information. Transliteration is also an open problem, as new names come up everyday and there is no absolute or one-to-one transliterated version for any name. Although direct orthographic mapping has implicitly or partially modelled the tone information via individual characters, the model nevertheless heavily depends on the availability of training data and could be skewed by the distribution of a certain homophone and thus precludes an acceptable transliteration alternative. We therefore propose to model the sound and tone together in *E2C*. In this way we attempt to deal with homophones more reasonably especially when the training data is limited. In this paper we report some work in progress and compare *E2C* in Cantonese and Mandarin Chinese.

Related work will be briefly reviewed in Section 2. Some characteristics of *E2C* will be discussed in Section 3. Work in progress will be reported in Section 4, followed by a conclusion with future work in Section 5.

2 Related Work

There are basically two categories of work on machine transliteration. First, various alignment models are used for acquiring transliteration lexicons from parallel corpora and other resources (e.g. Kuo and Li, 2008). Second, statistical models are built for transliteration. These models could be phoneme-based (e.g. Knight and Graehl, 1998), grapheme-based (e.g. Li *et al.*, 2004), hybrid (Oh and Choi, 2005), or based on phonetic (e.g. Tao *et al.*, 2006) and semantic (e.g. Li *et al.*, 2007) features.

Li *et al.* (2004) used a Joint Source-Channel Model under the direct orthographic mapping

[1] Mandarin names are transcribed in Hanyu Pinyin and Cantonese names are transcribed in Jyutping published by the Linguistic Society of Hong Kong.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 21–24,
Suntec, Singapore, 4 August 2009.

(DOM) framework, skipping the middle phonemic representation in conventional phoneme-based methods, and modelling the segmentation and alignment preferences by means of contextual n-grams of the transliteration units. Although DOM has implicitly modelled the tone choice, since a specific character has a specific tone, it nevertheless heavily relies on the availability of training data. If there happens to be a skewed distribution of a certain Chinese character, the model might preclude other acceptable transliteration alternatives. In view of the abundance of homophones in Chinese, and that sound-tone combination is important in names (i.e., names which sound "nice" are preferred to those which sound "monotonous"), we propose to model sound-tone combinations in transliteration more explicitly, using pinyin transcriptions to bridge the graphemic representation between English and Chinese. In addition, we also study the dialectal differences between transliteration in Mandarin Chinese and Cantonese, which is seldom addressed in past studies.

3 Some *E2C* Properties

3.1 Dialectal Differences

English and Chinese have very different phonological properties. A well cited example is a syllable initial /d/ may surface as in Baghdad 巴格達 *ba1-ge2-da2*, but the syllable final /d/ is not represented. This is true for Mandarin Chinese, but since ending stops like –p, –t and –k are allowed in Cantonese syllables, the syllable final /d/ in Baghdad is already captured in the last syllable of巴格達 *baa1-gaak3-daat6* in Cantonese.

Such phonological difference between Mandarin Chinese and Cantonese might also account for the observation that Cantonese transliterations often do not introduce extra syllables for certain consonant segments in the middle of an English name, as in Dickson, transliterated as 迪克遜 *di2-ke4-xun4* in Mandarin Chinese and 迪臣 *dik6-san4* in Cantonese.

3.2 Ambiguities from Homophones

The homophone problem is notorious in Chinese. As far as personal names are concerned, the "correctness" of transliteration is not clear-cut at all. For example, to transliterate the name Hilary into Chinese, based on Cantonese pronunciations, the following are possibilities amongst many others: (a) 希拉利 *hei1-laai1-lei6*, (b) 希拉莉 *hei1-laai1-lei6*, and (c) 希拉里 *hei1-laai1-lei5*.

The homophonous third character gives rise to multiple alternative transliterations in this example, where orthographically 利 *lei6*, 莉 *lei6* and 里 *lei5* are observed for "ry" in transliteration data. One cannot really say any of the combinations is "right" or "wrong", but perhaps only "better" or "worse". Such judgement is more cognitive than linguistic in nature, and apparently the tonal patterns play an important role in this regard. Hence naming is more of an art than a science, and automatic transliteration should avoid over-reliance on the training data and thus missing unlikely but good candidates.

4 Work in Progress

4.1 Datasets

A common set of 1,423 source English names and their transliterations[2] in Mandarin Chinese (as used by media in Mainland China) and Cantonese (as used by media in Hong Kong) were collected over the Internet. The names are mostly from soccer, entertainment, and politics. The data size is admittedly small compared to other existing transliteration datasets, but as a preliminary study, we aim at comparing the transliteration practice between Mandarin speakers and Cantonese speakers in a more objective way based on a common set of English names. The transliteration pairs were manually aligned, and the pronunciations for the Chinese characters were automatically looked up.

4.2 Preliminary Quantitative Analysis

	Cantonese	Mandarin
Unique name pairs	1,531	1,543
Total English segments	4,186	4,667
Unique English segments	969	727
Unique grapheme pairs	1,618	1,193
Unique seg-sound pairs	1,574	1,141

Table 1. Quantitative Aspects of the Data

As shown in Table 1, the average segment-name ratios (2.73 for Cantonese and 3.02 for Mandarin) suggest that Mandarin transliterations often use more syllables for a name. The much smaller number of unique English segments for Mandarin and the difference in token-type ratio of grapheme pairs (3.91 for Mandarin and 2.59 for Cantonese) further suggest that names are more consistently segmented and transliterated in Mandarin.

[2] Some names have more than one transliteration.

4.2.1 Graphemic Correspondence

Assume grapheme pair mappings are in the form $<e_k, \{c_{k1},c_{k2},\ldots,c_{kn}\}>$, where e_k stands for the kth unique English segment from the data, and $\{c_{k1},c_{k2},\ldots,c_{kn}\}$ for the set of n unique Chinese segments observed for it. It was found that n varies from 1 to 10 for Mandarin, with 34.9% of the distinct English segments having multiple grapheme mappings, as shown in Table 2. For Cantonese, n varies from 1 to 13, with 31.5% of the distinct English segments having multiple grapheme mappings. The proportion of multiple mappings is similar for Mandarin and Cantonese, but the latter has a higher percentage of English segments with 5 or more Chinese renditions. Thus Mandarin transliterations are relatively more "standardised", whereas Cantonese transliterations are graphemically more ambiguous.

n	Cantonese	Mandarin
>=5	5.3%	3.3%
4	4.0%	4.4%
3	6.2%	7.2%
2	16.0%	20.0%
1	68.5%	65.1%
Example	<le, {列, 利, 勒, 尼, 李, 歷, 烈, 爾, 理, 萊, 路, 里, 雷}>	<le, {列, 利, 勒, 歷, 爾, 理, 萊, 裏, 路, 雷}>

Table 2. Graphemic Ambiguity of the Data

4.2.2 Homophone Ambiguity (Sound Only)

Table 3 shows the situation with homophones (ignoring tones). For example, all five characters 利莉李里理 correspond to the Jyutping *lei*. Despite the tone difference, they are considered homophones in this section.

n	Cantonese	Mandarin
>=5	3.3%	1.9%
4	4.0%	2.5%
3	5.8%	5.7%
2	16.3%	20.7%
1	70.5%	69.2%
Example	<le, {ji, laak, lei, leoi, lik, lit, loi, lou, nei}>	<le, {er, lai, le, lei, li, lie, lu}>

Table 3. Homophone Ambiguity (Ignoring Tone)

Assume grapheme-sound pair mappings are in the form $<e_k, \{s_{k1},s_{k2},\ldots,s_{kn}\}>$, where e_k stands for the kth unique English segment, and $\{s_{k1},s_{k2},\ldots,s_{kn}\}$ for the set of n unique pronunciations (regardless of tone). For Mandarin, n varies from 1 to 7, with 30.8% of the distinct English segments having multiple sound mappings. For Cantonese, n varies from 1 to 9, with 29.5% of the distinct English segments having multiple sound mappings. Comparing with Table 2 above, the downward shift of the percentages suggests that much of the graphemic ambiguity is a result of the use of homophones, instead of a set of characters with very different pronunciations.

4.2.3 Homophone Ambiguity (Sound-Tone)

Table 4 shows the situation of homophones with both sound and tone taken into account. For example, the characters 利莉 all correspond to *lei6* in Cantonese, while 李里理 all correspond to *lei5*, and they are thus treated as two groups.

Assume grapheme-sound/tone pair mappings are in the form $<e_k, \{st_{k1},st_{k2},\ldots,st_{kn}\}>$, where e_k stands for the kth unique English segment, and $\{st_{k1},st_{k2},\ldots,st_{kn}\}$ for the set of n unique pronunciations (sound-tone combination). For Mandarin, n varies from 1 to 8, with 33.5% of the distinct English segments corresponding to multiple Chinese homophones. For Cantonese, n varies from 1 to 10, with 30.8% of the distinct English segments having multiple Chinese homophones.

n	Cantonese	Mandarin
>=5	4.1%	2.8%
4	4.8%	3.3%
3	6.1%	6.8%
2	15.8%	20.7%
1	69.2%	66.5%
Example	<le, {ji5, laak6, lei5, lei6, leoi4, lik6, lit6, loi4, lou6, nei4}>	<le, {er3, lai2, le4, lei2, li3, li4, lie4, lu4}

Table 4. Homophone Ambiguity (Sound-Tone)

The figures in Table 4 are somewhere between those in Table 2 and Table 3, suggesting that a considerable part of homophones used in the transliterations could be distinguished by tones. This supports our proposal of modelling tonal combination explicitly in *E2C*.

4.3 Method and Experiment

The Joint Source-Channel Model in Li *et al.* (2004) was adopted in this study. However, instead of direct orthographic mapping, we model the mapping between an English segment and the pronunciation in Chinese. Such a model is expected to have a more compact parameter space as individual Chinese characters for a certain English segment are condensed into homophones defined by a finite set of sounds and tones. The model could save on computational effort, and is less affected by any bias or sparseness of the data. We refer to this approach as SoTo hereafter.

Hence our approach with a bigram model is as follows:

$$P(E,ST) = P(e_1,e_2,...,e_k,st_1,st_2,...,st_k)$$
$$= P(<e_1,st_1>,<e_2,st_2>,...,<e_k,st_k>)$$
$$= \prod_{k=1}^{K} P(<e_k,st_k>|<e_{k-1},st_{k-1}>)$$

where E refers to the English source name and ST refers to the sound/tone sequence of the transliteration, while e_k and st_k refer to kth segment and its Chinese sound respectively. Homophones in Chinese are thus captured as a class in the phonetic transcription. For example, the expected Cantonese transliteration for Osborne is 奥斯邦尼 *ou3-si1-bong1-nei4*. Not only is it ranked first using this method, its homophonous variant 奥施邦尼 is within the top 5, thus benefitting from the grouping of the homophones, despite the relatively low frequency of <s,施>. This would be particularly useful for transliteration extraction and information retrieval.

Unlike pure phonemic modelling, the tonal factor is modelled in the pronunciation transcription. We do not go for phonemic representation from the source name as the transliteration of foreign names into Chinese is often based on the surface orthographic forms, e.g. the silent h in Beckham is pronounced to give 漢姆 *han4-mu3* in Mandarin and 咸 *haam4* in Cantonese.

Five sets of 50 test names were randomly extracted from the 1.4K names mentioned above for 5-fold cross validation. Training was done on the remaining data. Results were also compared with DOM. The Mean Reciprocal Rank (MRR) was used for evaluation (Kantor and Voorhees, 2000).

4.4 Preliminary Results

Method	Cantonese	Mandarin
DOM	0.2292	0.3518
SoTo	0.2442	0.3557

Table 5. Average System Performance

Table 5 shows the average results of the two methods. The figures are relatively low compared to state-of-the-art performance, largely due to the small datasets. Errors might have started to propagate as early as the name segmentation step. As a preliminary study, however, the potential of the SoTo method is apparent, particularly for Cantonese. A smaller model thus performs better, and treating homophones as a class could avoid over-reliance on the prior distribution of individual characters. The better performance for Mandarin data is not surprising given the less "standardised" Cantonese transliterations as discussed above. From the research point of view, it suggests more should be considered in addition to grapheme mapping for handling Cantonese data.

5 Future Work and Conclusion

Thus we have compared *E2C* between Mandarin Chinese and Cantonese, and discussed work in progress for our proposed SoTo method which more reasonably treats homophones and better models tonal patterns in transliteration. Future work includes testing on larger datasets, more in-depth error analysis, and developing better methods to deal with Cantonese transliterations.

Acknowledgements

The work described in this paper was substantially supported by a grant from City University of Hong Kong (Project No. 7002203).

References

Kantor, P.B. and Voorhees, E.M. (2000) The TREC-5 Confusion Track: Comparing Retrieval Methods for Scanned Text. *Information Retrieval, 2(2-3)*: 165-176.

Knight, K. and Graehl, J. (1998) Machine Transliteration. *Computational Linguistics, 24(4)*:599-612.

Kuo, J-S. and Li, H. (2008) Mining Transliterations from Web Query Results: An Incremental Approach. In *Proceedings of SIGHAN-6*, Hyderabad, India, pp.16-23.

Li, H., Zhang, M. and Su, J. (2004) A Joint Source-Channel Model for Machine Transliteration. In *Proceedings of the 42nd Annual Meeting of ACL*, Barcelona, Spain, pp.159-166.

Li, H., Sim, K.C., Kuo, J-S. and Dong, M. (2007) Semantic Transliteration of Personal Names. In *Proceedings of the 45th Annual Meeting of ACL*, Prague, Czech Republic, pp.120-127.

Oh, J-H. and Choi, K-S. (2005) An Ensemble of Grapheme and Phoneme for Machine Transliteration. In R. Dale *et al.* (Eds.), *Natural Language Processing – IJCNLP 2005*. Springer, LNAI Vol. 3651, pp.451-461.

Tao, T., Yoon, S-Y., Fister, A., Sproat, R. and Zhai, C. (2006) Unsupervised Named Entity Transliteration Using Temporal and Phonetic Correlation. In *Proceedings of EMNLP 2006*, Sydney, Australia, pp.250-257.

Virga, P. and Khudanpur, S. (2003) Transliteration of Proper Names in Cross-lingual Information Retrieval. In *Proceedings of the ACL2003 Workshop on Multilingual and Mixed-language Named Entity Recognition*.

Capturing Errors in Written Chinese Words

Chao-Lin Liu[1] Kan-Wen Tien[2] Min-Hua Lai[3] Yi-Hsuan Chuang[4] Shih-Hung Wu[5]
[1-4]National Chengchi University, [5]Chaoyang University of Technology, Taiwan
{[1]chaolin, [2]96753027, [3]95753023, [4]94703036}@nccu.edu.tw, [5]shwu@cyut.edu.tw

Abstract

A collection of 3208 reported errors of Chinese words were analyzed. Among which, 7.2% involved rarely used character, and 98.4% were assigned common classifications of their causes by human subjects. In particular, 80% of the errors observed in writings of middle school students were related to the pronunciations and 30% were related to the compositions of words. Experimental results show that using intuitive Web-based statistics helped us capture only about 75% of these errors. In a related task, the Web-based statistics are useful for recommending incorrect characters for composing test items for "incorrect character identification" tests about 93% of the time.

1 Introduction

Incorrect writings in Chinese are related to our understanding of the cognitive process of reading Chinese (e.g., Leck et al., 1995), to our understanding of why people produce incorrect characters and our offering corresponding remedies (e.g., Law et al., 2005), and to building an environment for assisting the preparation of test items for assessing students' knowledge of Chinese characters (e.g., Liu and Lin, 2008).

Chinese characters are composed of smaller parts that can carry phonological and/or semantic information. A Chinese word is formed by Chinese characters. For example, 新加坡 (Singapore) is a word that contains three Chinese characters. The left (土) and the right (皮) part of 坡, respectively, carry semantic and phonological information. Evidences show that production of incorrect characters are related to either phonological or the semantic aspect of the characters.

In this study, we investigate several issues that are related to incorrect characters in Chinese words. In Section 2, we present the sources of the reported errors. In Section 3, we analyze the causes of the observed errors. In Section 4, we explore the effectiveness of relying on Web-based statistics to correct the errors. The current results are encouraging but demand further improvements. In Section 5, we employ Web-based statistics in the process of assisting teachers to prepare test items for assessing students' knowledge of Chinese characters. Experimental results showed that our method outperformed the one reported in (Liu and Lin, 2008), and captured the best candidates for incorrect characters 93% of the time.

2 Data Sources

We obtained data from three major sources. A list that contains 5401 characters that have been believed to be sufficient for everyday lives was obtained from the Ministry of Education (MOE) of Taiwan, and we call the first list the **Clist**, henceforth. We have two lists of words, and each word is accompanied by an incorrect way to write certain words. The first list is from a book published by MOE (MOE, 1996). The MOE provided the correct words and specified the incorrect characters which were mistakenly used to replace the correct characters in the correct words. The second list was collected, in 2008, from the written essays of students of the seventh and the eighth grades in a middle school in Taipei. The incorrect words were entered into computers based on students' writings, ignoring those characters that did not actually exist and could not be entered.

We will call the first list of incorrect words the **Elist**, and the second the **Jlist** from now on. Elist and Jlist contain, respectively, 1490 and 1718 entries. Each of these entries contains a correct word and the incorrect character. Hence, we can reconstruct the incorrect words easily. Two or more different ways to incorrectly write the same words were listed in different entries and considered as two entries for simplicity of presentation.

3 Error Analysis of Written Words

Two subjects, who are native speakers of Chinese and are graduate students in Computer Science, examined Elist and Jlist and categorized the causes of errors. They compared the incorrect characters with the correct characters to determine whether the errors were **pronunciation-related** or semantic-related. Referring to an error as being "semantic-related" is ambiguous. Two characters might not contain the same semantic part, but are still semantically related. In this study, we have not considered this factor. For this reason we refer to the errors that are related to the sharing of semantic parts in characters as **composition-related**.

It is interesting to learn that native speakers had a high consensus about the causes for the observed errors, but they did not always agree. Hence, we studied the errors that the two subjects had agreed categorizations. Among the 1490 and 1718 words in Elist and Jlist, respectively, the two human subjects had consensus over causes of 1441 and 1583 errors.

The statistics changed when we disregarded errors that involved characters not included in Clist. An error would be ignored if either the correct or the incorrect character did not belong to the Clist. It is possible for students to write such rarely used characters in an incorrect word just by coincidence.

After ignoring the rare characters, there were 1333 and 1645 words in Elist and Jlist, respectively. The subjects had consensus over the categories for 1285

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 25–28,
Suntec, Singapore, 4 August 2009.

Table 1. Error analysis for Elist and Jlist

	C	*P*	*C&P*	*NE*	*D*
Elist	66.09%	67.21%	37.13%	0.23%	3.60%
Jlist	30.70%	**79.88%**	20.91%	2.43%	7.90%

and 1515 errors in Elist and Jlist, respectively.

Table 1 shows the percentages of five categories of errors: ***C*** for the composition-related errors, ***P*** for the pronunciation-related errors, ***C&P*** for the intersection of *C* and *P*, ***NE*** for those errors that belonged to neither *C* nor *P*, and ***D*** for those errors that the subjects disagreed on the error categories. There were, respectively, 505 composition-related and 1314 pronunciation-related errors in Jlist, so we see 30.70% (=505/1645) and 79.88% (=1314/1645) in the table. Notice that *C&P* represents the intersection of *C* and *P*, so we have to deduct C&P from the sum of *C*, *P*, *NE*, and *D* to find the total probability, namely 1.

It is worthwhile to discuss the implication of the statistics in Table 1. For the Jlist, similarity between pronunciations accounted for nearly 80% of the errors, and the ratio for the errors that are related to compositions and pronunciations is 1:2.6. In contrast, for the Elist, the corresponding ratio is almost 1:1. The Jlist and Elist differed significantly in the ratios of the error types. It was assumed that the dominance of pronunciation-related errors in electronic documents was a result of the popularity of entering Chinese with pronunciation-based methods. The ratio for the Jlist challenges this popular belief, and indicates that even though the errors occurred during a writing process, rather than typing on computers, students still produced more pronunciation-related errors than composition-related errors. Distribution over error types is not as related to input method as one may have believed. Nevertheless, the observation might still be a result of students being so used to entering Chinese text with pronunciation-based method that the organization of their mental lexicons is also pronunciation related. The ratio for the Elist suggests that editors of the MOE book may have chosen the examples with a special viewpoint in their minds – balancing the errors due to pronunciation and composition.

4 Reliability of Web-based Statistics

In this section, we examine the effectiveness of using Web-based statistics to differentiate correct and incorrect characters. The abundant text material on the Internet gives people to treat the Web as a corpus (e.g., webascorpus.org). When we send a query to Google, we will be informed of the number of pages (NOPs) that possibly contain relevant information. If we put the query terms in quotation marks, we should find the web pages that literally contain the query terms. Hence, it is possible for us to compare the NOPs for two competing phrases for guessing the correct way of writing. At the time of this writing, Google found 107000 and 3220 pages, respectively, for "strong tea" and "powerful tea". (When conducting such advanced searches with Google, the quotation marks are needed to ensure the adjacency of individual words.) Hence, "strong" appears to be a better choice to go with "tea". How does this strategy serve for learners of Chinese?

Table 2. Reliability of Web-based statistics

		Trad		Twn+Trad	
		Comp	Pron	Comp	Pron
Elist	C	**73.12%**	73.80%	69.92%	68.72%
	A	**4.58%**	3.76%	3.83%	3.76%
	I	**22.30%**	22.44%	26.25%	27.52%
Jlist	C	76.59%	74.98%	69.34%	65.87%
	A	2.26%	3.97%	2.47%	5.01%
	I	21.15%	21.05%	28.19%	29.12%

We verified this strategy by sending the words in both the Elist and the Jlist to Google to find the NOPs. We can retrieve the NOPs from the documents returned by Google, and compare the NOPs for the correct and the incorrect words to evaluate the strategy. Again, we focused on those in the 5401 words that the human subjects had consensus about their error types. Recall that we have 1285 and 1515 such words in Elist and Jlist, respectively. As the information available on the Web changes all the time, we also have to note that our experiments were conducted during the first half of March 2009. The queries were submitted at reasonable time intervals to avoid Google's treating our programs as malicious attackers.

Table 2 shows the results of our investigation. We considered that we had a correct result when we found that the NOP for the correct word larger than the NOP for the incorrect word. If the NOPs were equal, we recorded an ambiguous result; and when the NOP for the incorrect word is larger, we recorded an incorrect event. We use '**C**', '**A**', and '**I**' to denote "correct", "ambiguous", and "incorrect" events in Table 2.

The column headings of Table 2 show the setting of the searches with Google and the set of words that were used in the experiments. We asked Google to look for information from web pages that were encoded in traditional Chinese (denoted **Trad**). We could add another restriction on the source of information by asking Google to inspect web pages from machines in Taiwan (denoted **Twn+Trad**). We were not sure how Google determined the languages and locations of the information sources, but chose to trust Google. The headings "**Comp**" and "**Pron**" indicate whether the words whose error types were composition and pronunciation-related, respectively.

Table 2 shows eight distributions, providing experimental results that we observed under different settings. The distribution printed in bold face showed that, when we gathered information from sources that were encoded in traditional Chinese, we found the correct words 73.12% of the time for words whose error types were related to composition in Elist. Under the same experimental setting, we could not judge the correct word 4.58% of the time, and would have chosen an incorrect word 22.30% of the time.

Statistics in Table 2 indicate that web statistics is not a very reliable factor to judge the correct words. The average of the eight numbers in the 'C' rows is only 71.54% and the best sample is 76.59%, suggest-

ing that we did not find the correct words frequently. We would made incorrect judgments 24.75% of the time. The statistics also show that it is almost equally difficult to find correct words for errors that are composition and pronunciation related. In addition, the statistics reveal that choosing more features in the advanced search affected the final results. Using "Trad" offered better results in our experiments than using "Twn+Trad". This observation may arouse a perhaps controversial argument. Although Taiwan has proclaimed to be the major region to use traditional Chinese, their web pages might not have used as accurate Chinese as web pages located in other regions.

We have analyzed the reasons for why using Web-based statistics did not find the correct words. Frequencies might not have been a good factor to determine the correctness of Chinese. However, the myriad amount of data on the Web should have provided a better performance. Google's rephrasing our submitted queries is an important factor, and, in other cases, incorrect words were more commonly used.

5 Facilitating Test Item Authoring

Incorrect character correction is a very popular type of test in Taiwan. There are simple test items for young children, and there are very challenging test items for the competitions among adults. Finding an attractive incorrect character to replace a correct character to form a test item is a key step in authoring test items.

We have been trying to build a software environment for assisting the authoring of test items for incorrect character correction (Liu and Lin, 2008, Liu et al., 2009). It should be easy to find a lexicon that contains pronunciation information about Chinese characters. In contrast, it might not be easy to find visually similar Chinese characters with computational methods. We expanded the original Cangjie codes (**OCC**), and employed the expanded Cangjie codes (**ECC**) to find visually similar characters (Liu and Lin, 2008).

With a lexicon, we can find characters that can be pronounced in a particular way. However, this is not enough for our goal. We observed that there were different symptoms when people used incorrect characters that are related to their pronunciations. They may use characters that could be pronounced exactly the same as the correct characters. They may also use characters that have the same pronunciation and different tones with the correct character. Although relatively infrequently, people may use characters whose pronunciations are similar to but different from the pronunciation of the correct character.

As Liu and Lin (2008) reported, replacing OCC with ECC to find visually similar characters could increase the chances to find similar characters. Yet, it was not clear as to which components of a character should use ECC.

5.1 Formalizing the Extended Cangjie Codes

We analyzed the OCCs for all the words in Clist to determine the list of basic components. We treated a Cangjie basic symbol as if it was a word, and computed the number of occurrences of n-grams based on the OCCs of the words in Clist. Since the OCC for a character contains at most five symbols, the longest n-grams are 5-grams. Because the reason to use ECC was to find common components in characters, we disregarded n-grams that repeated no more than three times. In addition, the n-grams that appeared more than three times might not represent an actual component in Chinese characters. Hence, we also removed such n-grams from the list of our basic components. This process naturally made our list include radicals that are used to categorize Chinese characters in typical printed dictionaries. The current list contains 794 components, and it is possible to revise the list of basic components in our work whenever necessary.

After selecting the list of basic components with the above procedure, we encoded the words in Elist with our list of basic components. We adopted the 12 ways that Liu and Lin (2008) employed to decompose Chinese characters. There are other methods for decomposing Chinese characters into components. Juang et al. (2005) and the research team at the Sinica Academia propose 13 different ways for decomposing characters.

5.2 Recommending Incorrect Alternatives

With a dictionary that provides the pronunciation of Chinese characters and the improved ECC encodings for words in the Elist, we can create lists of candidate characters for replacing a specific correct character in a given word to create a test item for incorrect character correction.

There are multiple strategies to create the candidate lists. We may propose the candidate characters because their pronunciations have the **s**ame **s**ound and the **s**ame **t**one with those of the correct character (denoted ***SSST***). Characters that have **s**ame **s**ounds and **d**ifferent **t**ones (***SSDT***), characters that have si**m**ilar **s**ounds and **s**ame **t**ones (***MSST***), and characters that have si**m**ilar **s**ounds and **d**ifferent **t**ones (***MSDT***) can be considered as candidates as well. It is easy to judge whether two Chinese characters have the same tone. In contrast, it is not trivial to define "similar" sound. We adopted the list of similar sounds that was provided by a psycholinguistic researcher (Dr. Chia-Ying Lee) at the Sinica Academia.

In addition, we may propose characters that look similar to the correct character. Two characters may look similar for two reasons. They may contain the same components, or they contain the same **r**adical and have the same total number of **s**trokes (***RS***). When two characters contain the same component, the shared component might or might not locate at the same position within the bounding boxes of characters.

In an authoring tool, we could recommend a limited number of candidate characters for replacing the correct character. We tried two strategies to compare and choose the visually similar characters. The first strategy (denoted ***SC1***) gave a higher score to the shared component that located at the same location in the two characters being compared. The second strat-

Table 3. Incorrect characters were contained and ranked high in the recommended lists

	SC1	*SC2*	*RS*	*SSST*	*SSDT*	*MSST*	*MSDT*	*Comp*	*Pron*	*Both*
Elist	73.92%	76.08%	4.08%	91.64%	18.39%	3.01%	1.67%	81.97%	99.00%	93.37%
Jlist	67.52%	74.65%	6.14%	92.16%	20.24%	4.19%	3.58%	77.62%	99.32%	97.29%
Elist	3.25	2.91	1.89	2.30	1.85	2.00	1.58			
Jlist	2.82	2.64	2.19	3.72	2.24	2.77	1.16			

egy (***SC2***) gave the same score to any shared component even if the component did not reside at the same location in the characters. When there were more than 20 characters that receive nonzero scores, we chose to select at most 20 characters that had leading scores as the list of recommended characters.

5.3 Evaluating the Recommendations

We examined the usefulness of these seven categories of candidates with errors in Elist and Jlist. The first set of evaluation (the inclusion tests) checked only whether the lists of recommended characters contained the incorrect character in our records. The second set of evaluation (the ranking tests) was designed for practical application in computer assisted item generation. Only for those words whose actual incorrect characters were included in the recommended list, we replaced the correct characters in the words with the candidate incorrect characters, submitted the incorrect words to Google, and ordered the candidate characters based on their NOPs. We then recorded the ranks of the incorrect characters among all recommended characters.

Since the same character may appear simultaneously in *SC1*, *SC2*, and *RS*, we computed the union of these three sets, and checked whether the incorrect characters were in the union. The inclusion rate is listed under ***Comp***. Similarly, we computed the union for *SSST*, *SSDT*, *MSST*, and *MSDT*, checked whether the incorrect characters were in the union, and recorded the inclusion rate under ***Pron***. Finally, we computed the union of the lists created by the seven strategies, and recorded the inclusion rate under ***Both***.

The second and the third rows of Table 3 show the results of the inclusion tests. The data show the percentage of the incorrect characters being included in the lists that were recommended by the seven strategies. Notice that the percentages were calculated with different denominators. The number of composition-related errors was used for *SC1*, *SC2*, *RS*, and *Comp* (e.g. 505 that we mentioned in Section 3 for the Jlist); the number of pronunciation-related errors for *SSST*, *SSDT*, *MSST*, *MSDT*, and *Pron* (e.g., 1314 mentioned in Section 3 for the Jlist); the number of either of these two errors for *Both* (e.g., 1475 for Jlist).

The results recorded in Table 3 show that we were able to find the incorrect character quite effectively, achieving better than 93% for both Elist and Jlist. The statistics also show that it is easier to find incorrect characters that were used for pronunciation-related problems. Most of the pronunciation-related problems were misuses of characters that had exactly the same pronunciations with the correct characters. Unexpected confusions, e.g., those related to pronunciations in Chinese dialects, were the main for the failure to capture the pronunciation-related errors. SSDT is a crucial complement to SSST. There is still room to improve our methods to find confusing characters based on their compositions. We inspected the list generated by SC1 and SC2, and found that, although SC2 outperformed SC1 on the inclusion rate, SC1 and SC2 actually generated complementary lists and should be used together. The inclusion rate achieved by the RS strategy was surprisingly high.

The fourth and the fifth rows of Table 3 show the effectiveness of relying on Google to rank the candidate characters for recommending an incorrect character. The rows show the average ranks of the included cases. The statistics show that, with the help of Google, we were able to put the incorrect character on top of the recommended list when the incorrect character was included. This allows us to build an environment for assisting human teachers to efficiently prepare test items for incorrect character identification.

6 Summary

The analysis of the 1718 errors produced by real students show that similarity between pronunciations of competing characters contributed most to the observed errors. Evidences show that the Web statistics are not very reliable for differentiating correct and incorrect characters. In contrast, the Web statistics are good for comparing the attractiveness of incorrect characters for computer assisted item authoring.

Acknowledgements

This research has been funded in part by the National Science Council of Taiwan under the grant NSC-97-2221-E-004-007-MY2. We thank the anonymous reviewers for invaluable comments, and more responses to the comments are available in (Liu et al. 2009).

References

D. Juang, J.-H. Wang, C.-Y. Lai, C.-C. Hsieh, L.-F. Chien, J.-M. Ho. 2005. Resolving the unencoded character problem for Chinese digital libraries, *Proc. of the 5th ACM/IEEE Joint Conf. on Digital Libraries*, 311–319.

S.-P. Law, W. Wong, K. M. Y. Chiu. 2005. Whole-word phonological representations of disyllabic words in the Chinese lexicon: Data from acquired dyslexia, *Behavioural Neurology*, **16**, 169–177.

K. J. Leck, B. S. Weekes, M. J. Chen. 1995. Visual and phonological pathways to the lexicon: Evidence from Chinese readers, *Memory & Cognition*, **23**(4), 468–476.

C.-L. Liu et al. 2009. Phonological and logographic influences on errors in written Chinese words, *Proc. of the 7th Workshop on Asian Language Resources*, 47th ACL.

C.-L. Liu, J.-H. Lin. 2008. Using structural information for identifying similar Chinese characters, *Proc. of the 46th ACL*, short papers, 93−96.

MOE. 1996. *Common Errors in Chinese Writings* (常用國字辨似), Ministry of Education, Taiwan.

A Novel Word Segmentation Approach for Written Languages with Word Boundary Markers

Han-Cheol Cho[†], Do-Gil Lee[§], Jung-Tae Lee[§], Pontus Stenetorp[†], Jun'ichi Tsujii[†] and Hae-Chang Rim[§]
[†]Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan
[§]Dept. of Computer & Radio Communications Engineering, Korea University, Seoul, Korea
{hccho,pontus,tsujii}@is.s.u-tokyo.ac.jp, {dglee,jtlee,rim}@nlp.korea.ac.kr

Abstract

Most NLP applications work under the assumption that a user input is error-free; thus, word segmentation (WS) for written languages that use word boundary markers (WBMs), such as spaces, has been regarded as a trivial issue. However, noisy real-world texts, such as blogs, e-mails, and SMS, may contain spacing errors that require correction before further processing may take place. For the Korean language, many researchers have adopted a traditional WS approach, which eliminates all spaces in the user input and re-inserts proper word boundaries. Unfortunately, such an approach often exacerbates the word spacing quality for user input, which has few or no spacing errors; such is the case, because a perfect WS model does not exist. In this paper, we propose a novel WS method that takes into consideration the initial word spacing information of the user input. Our method generates a better output than the original user input, even if the user input has few spacing errors. Moreover, the proposed method significantly outperforms a state-of-the-art Korean WS model when the user input initially contains less than 10% spacing errors, and performs comparably for cases containing more spacing errors. We believe that the proposed method will be a very practical pre-processing module.

1 Introduction

Word segmentation (WS) has been a fundamental research issue for languages that do not have word boundary markers (WBMs); on the contrary, other languages that do have WBMs have regarded the issue as a trivial task. Texts segmented with such WBMs, however, could contain a human writer's intentional or un-intentional spacing errors; and even a few spacing errors can cause error-propagation for further NLP stages.

For written languages that have WBMs, such as for the Korean language, the majority of recent research has been based on a traditional WS approach (Nakagawa, 2004). The first step of the traditional approach is to eliminate all spaces in the user input, and then re-locate the proper places to insert WBMs. One state-of-the-art Korean WS model (Lee et al., 2007) is known to achieve a performance of 90.31% word-unit precision, which is comparable with other WS models for the Chinese or Japanese language.

Still, there is a downside to the evaluation method. If the user input has a few or no spacing errors, traditional WS models may cause more spacing errors than it correct because they produce the same output regardless the word spacing states of the user input.

In this paper, we propose a new WS method that takes into account the word spacing information from the user input. Our proposed method first generates the best word spacing states for the user input by using a traditional WS model; however the method does not immediately apply the output. Secondly, the method estimates a threshold based on the word spacing quality of the user input. Finally, the method uses the new word spacing states that have probabilities that are higher than the threshold.

The most important contribution of the proposed method is that, for most cases, the method generates an output that is better than the user input. The experimental results show that the proposed method produces a better output than the user input even if the user input has less than 1% spacing errors in terms of the *character-unit precision*. Moreover, the proposed method outperforms (Lee et al., 2007) significantly, when the

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 29–32,
Suntec, Singapore, 4 August 2009.

user input initially contains less than 10% spacing errors, and even performs comparably, when the input contains more than 10% errors. Based on these results, we believe that the proposed method would be a very practical pre-processing module for other NLP applications.

The paper is organized as follows: Section 2 explains the proposed method. Section 3 shows the experimental results. Finally, the last section describes the contributions of the proposed method.

2 The Proposed Method

The proposed method consists of three steps: a baseline WS model, confidence and threshold estimation, and output optimization. The following sections will explain the steps in detail.

2.1 Baseline Word Segmentation Model

We use the tri-gram Hidden Markov Model (HMM) of (Lee et al., 2007) as the baseline WS model; however, we adopt the Maximum Likelihood (ML) decoding strategy to independently find the best word spacing states. ML-decoding allows us to directly compare each output to the threshold. There is little discrepancy in accuracy when using ML-decoding, as compared to Viterbi-decoding, as mentioned in (Merialdo, 1994).[1]

Let $o_{1,n}$ be a sequence of n-character user input without WBMs, x_t be the best word spacing state for o_t where $1 \leq t \leq n$. Assume that x_t is either 1 (space after o_t) or 0 (no space after o_t). Then each best word spacing state $\hat{x}_t$ for all t can be found by using Equation 1.

$$\hat{x_t} = \underset{i \in (0,1)}{\operatorname{argmax}} P(x_t = i | o_{1,n}) \quad (1)$$

$$= \underset{i \in (0,1)}{\operatorname{argmax}} P(o_{1,n}, x_t = i) \quad (2)$$

$$= \underset{i \in (0,1)}{\operatorname{argmax}} \sum_{x_{t-2}, x_{t-1}} P(x_t = i | x_{t-2}, o_{t-1}, x_{t-1}, o_t) \times \sum_{x_{t-1}} P(o_{t+1} | o_{t-1}, x_{t-1}, o_t, x_t = i) \times \sum_{x_{t+1}} P(o_{t+2} | o_t, x_t = i, o_{t+1}, x_{t+1}) \quad (3)$$

Equation 2 is derived by applying the Bayes' rule and by eliminating the constant denominator. Moreover, the equation is simplified, as is Equation 3, by using the Markov assumption, and by eliminating the constant parts. Every part of Equation 3 can be calculated by adding the probabilities of all possible combinations of x_{t-2}, x_{t-1}, x_{t+1} and x_{t+2} values.

The model is trained by using the relative frequency information of the training data, and a smoothing technique is applied to relieve the data-sparseness problem which is the linear interpolation of n-grams that are used in (Lee et al., 2007).

2.2 Confidence and Threshold Estimation

We set a variable threshold that is proportional to the word spacing quality of the user input, *Confidence*. Formally, we can define the threshold T as a function of a confidence C, as in Equation 4.

$$T = f(C) \quad (4)$$

Then, we define the confidence as is done in Equation 5. Because calculating such a variable is impossible, we estimate the value by substituting the word spacing states produced by the baseline WS model, $x_{1,n}^{WS}$, with the correct word spacing states, $x_{1,n}^{correct}$, as is done in Equation 6. This estimation is based on the assumption that the word spacing states of the WS model is sufficiently similar to the correct word spacing states in the ***character-unit precision***.[2]

$$C = \frac{\text{\# of } x_t^{input} \text{ same to } x_t^{correct}}{\text{\# of } x_t^{input}} \quad (5)$$

$$\approx \frac{\text{\# of } x_t^{input} \text{ same to } x_t^{WS}}{\text{\# of } x_t^{input}} \quad (6)$$

$$\approx \sqrt[n]{\prod_{k=1}^{n} P(x_k^{input} | o_{1,n})} \quad (7)$$

To handle the estimation error for short sentences, we use the probability generating word spacing states of the user input with the length normalization as shown in Equation 7.

Figure 1 shows that the estimated confidence of Equation 7 is almost linearly proportional to the true confidence of Equation 5, thus suggesting that the threshold T can be defined as a function of the estimated confidence of Equation 7.[3]

[1] In the preliminary experiment, Viterbi-decoding showed a 0.5% higher word-unit precision.

[2] In the experiment with the development data, the baseline WS model shows about 97% character-unit precision.

[3] The development data is generated by randomly introducing spacing errors into correctly spaced sentences. We think that this reflects various intentional and un-intentional error patterns of individuals.

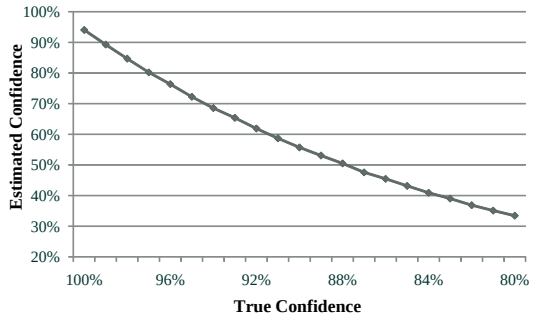


Figure 1: The relationship between estimated confidence and true confidence

To keep the focus on the research subject of this paper, we simply assume $f(x) = x$ as in Equation 8, for the threshold function f.

$$T \approx f(C) = C \tag{8}$$

In the experimental results, we confirm that even this simple threshold function can be helpful in improving the performance of the proposed method against traditional WS models.

2.3 Output Optimization

After completing the two steps described in Section 2.1 and 2.2, we have acquired the new spacing states for the user input generated by the baseline WS model, and the threshold measuring the word spacing quality of the user input.

The proposed method only applies a part of the new word spacing states to the user input, which have probabilities that are higher than the threshold; further the method discards the other new word spacing states that have probabilities that are lower than the threshold. By rejecting the unreliable output of the baseline WS model in this way, the proposed method can effectively improve the performance when the user input contains a relatively small number of spacing errors.

3 Experimental Results

Two types of experiments have been performed. In the first experiment, we investigate the level of performance improvement based on different settings of the user input's word spacing error rate. Because it is nearly impossible to obtain enough test data for any error rate, we generate pseudo test data in the same way that we generate development data.[4] In the second experiment, we attempt figuring out whether the proposed method really improves the word spacing quality of the user input in a real-world setting.

3.1 Performance Improvement according to the Word Spacing Error Rate of User Input

For the first experiment, we use the Sejong corpus[5] from 1998-1999 (1,000,000 Korean sentences) for the training data, and ETRI corpus (30,000 sentences) for the test data (ETRI, 1999). To generate the test data that have spacing errors, we make twenty one copies of the test data and randomly insert spacing errors from 0% to 20% in the same way in which we made the development data. We feel that this strategy can model both the intentional and un-intentional human error patterns.

In Figure 2, the x-axis indicates the word spacing error rate of the user input in terms of the character-unit precision, and the y-axis shows the word-unit precision of the output. Each graph depicts the word-unit precision of the test corpus, a state-of-the-art Korean WS model (Lee et al., 2007), the baseline WS model, and the proposed method.

Although Lee's model is known to perform comparably with state-of-the-art Chinese and Japanese WS models, it does not necessarily suggest that the word spacing quality of the model's output is better than the user input. In Figure 2, Lee's model exacerbates the user input when it has spacing errors that are lower than 3%.

The proposed method, however, produces a better output, even if the user input has 1% spacing errors. Moreover, the proposed method shows a considerably better performance within the 10% spacing error range, as compared to Lee's model, although the baseline WS model itself does not outperforms Lee's model. The performance improvement in this error range is fairly significant because we found that the spacing error rate of texts collected for the second experiment was about 9.1%.

3.2 Performance Comparison with Web Text having Usual Error Rate

In the second experiment, we attempt finding out whether the proposed method can be beneficial under real-world circumstances. Web texts, which consist of 1,000 erroneous sentences from famous

[4] See Footnote 3.

[5] Details available at: http://www.sejong.or.kr/eindex.php

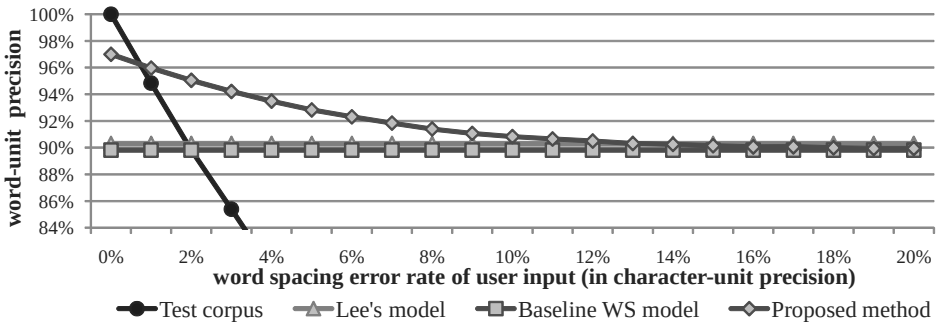


Figure 2: Performance improvement according to the word spacing error rate of user input

Method	Web Text
Test Corpus	70.89%
Lee's Model	70.45%
Baseline WS Model	69.13%
Proposed Method	73.74%

Table 1: Performance comparison with Web text

Web portals and personal blogs, were collected and used as the test data. Since the test data tend to have a similar error rate to the narrow standard deviation, we computed the overall performance over the average word spacing error rate, which is 9.1%. The baseline WS model is trained on the Sejong corpus, described in Section 3.1.

The test result is shown in Table 1. The overall performance of Lee's model, the baseline WS model and the proposed method decreased by roughly 18%. We hypothesize that the performance degradation probably results from the spelling errors of the test data, and the inconsistencies that exist between the training data and the test data. However, the proposed method still improves the word spacing quality of the user input by 3%, while the two traditional WS models degrades the quality. Such a result indicates that the proposed method is effective for real-world environments, as we had intended. Furthermore, we also believe that the performance can be improved if a proper training corpus is provided, or if a spelling correction method is integrated.

4 Conclusion

In this paper, we proposed a new WS method that uses the word spacing information of the user input, for languages with WBMs. By utilizing the user input, the proposed method effectively refines the output of the baseline WS model and improves the overall performance.

The most important contribution of this work is that it produces an output that is better than the user input even if it contains few spacing errors. Therefore, the proposed method can be applied as a pre-processing module for practical NLP applications without introducing a risk that would generate a worse output than the user input. Moreover, the performance is notably better than a state-of-the-art Korean WS model (Lee et al., 2007) within the 10% spacing error range, which human writers seldom exceed. It also performs comparably, even if the user input contains more than 10% spacing errors.

5 Acknowledgment

This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan) and Special Coordination Funds for Promoting Science and Technology (MEXT, Japan).

References

ETRI. 1999. Pos-tag guidelines. Technical report. *Electronics and Telecomminications Research Institute*.

Do-Gil Lee, Hae-Chang Rim, and Dongsuk Yook. 2007. Automatic Word Spacing Using Probabilistic Models Based on Character n-grams. *IEEE Intelligent Systems*, 22(1):28–35.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Comput. Linguist.*, 20(2):155–171.

Tetsuji Nakagawa. 2004. Chinese and Japanese word segmentation using word-level and character-level information. In *COLING '04*, page 466, Morristown, NJ, USA. Association for Computational Linguistics.

Part of Speech Tagger for Assamese Text

Navanath Saharia
Department of CSE
Tezpur University
India - 784028

Dhrubajyoti Das
Department of CSE
Tezpur University
India - 784028

Utpal Sharma
Department of CSE
Tezpur University
India - 784028

Jugal Kalita
Department of CS
University of Colorado
Colorado Springs - 80918

{nava_tu,dhruba_it06,utpal}@tezu.ernet.in
kalita@eas.uccs.edu

Abstract

Assamese is a morphologically rich, agglutinative and relatively free word order Indic language. Although spoken by nearly 30 million people, very little computational linguistic work has been done for this language. In this paper, we present our work on part of speech (POS) tagging for Assamese using the well-known Hidden Markov Model. Since no well-defined suitable tagset was available, we develop a tagset of 172 tags in consultation with experts in linguistics. For successful tagging, we examine relevant linguistic issues in Assamese. For unknown words, we perform simple morphological analysis to determine probable tags. Using a manually tagged corpus of about 10000 words for training, we obtain a tagging accuracy of nearly 87% for test inputs.

1 Introduction

Part of Speech (POS) tagging is the process of marking up words and punctuation characters in a text with appropriate POS labels. The problems faced in POS tagging are many. Many words that occur in natural language texts are not listed in any catalog or lexicon. A large percentage of words also show ambiguity regarding lexical category.

The challenges of our work on POS tagging for Assamese, an Indo-European language, are compounded by the fact that very little prior computational linguistic exists for the language, though it is a national language of India and spoken by over 30 million people. Assamese is a morphologically rich, free word order, inflectional language. Although POS tagged annotated corpus for some of the Indian languages such as Hindi, Bengali, and Telegu (SPSAL, 2007) have become available lately, a POS tagged corpus for Assamese was unavailable till we started creating one for the work presented in this paper. Another problem was that a clearly defined POS tagset for Assamese was unavailable to us. As a part of the work reported in this paper, we have developed a tagset consisting of 172 tags, using this tagset we have manually tagged a corpus of about ten thousand Assamese words.

In the next section we provide a brief relevant linguistic background of Assamese. Section 3 contains an overview of work on POS tagging. Section 4 describes our experimental setup. In Section 5, we analyse the result of our work and compare the performance with other models. Section 6 concludes this paper.

2 Linguistic Characteristics of Assamese

In Assamese, secondary forms of words are formed through three processes: affixation, derivation and compounding. Affixes play a very important role in word formation. Affixes are used in the formation of relational nouns and pronouns, and in the inflection of verbs with respect to number, person, tense, aspect and mood. For example, Table 1 shows how a relational noun দেউতা (*deutA*: father) is inflected depending on number and person (Goswami, 2003). Though Assamese is relatively free word order, yet the predominant word order is *subject-object-verb (SOV)*.

The following paragraphs describe just a few of the many characteristics of Assamese text that make the tagging task complex.

- Depending on the context, even a common word may have different POS tags. For example: If কাৰণে (*kArane*), দৰে (*dare*), নিমিত্তে (*nimitte*), হেতু (*hetu*), etc., are placed after pronominal adjective, they are considered conjunction and if placed after

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 33–36,
Suntec, Singapore, 4 August 2009.

Table 1: Personal definitives are inflected on person and number

Person	Singular	Plural
1^{st} প্রথম	My father মোৰ দেউতা *mor deutA*	Our father আমাৰ দেউতা *aAmAr deutA*
2^{nd} মান্য মধ্যম	Your father তোমাৰ দেউতাৰা *tomAr deutArA*	Your father তোমালোকৰ দেউতাৰা *tomAlokar deutArA*
2^{nd}, Familiar তুচ্ছ মধ্যম	Your father তোৰ দেউতাৰ *tor deutAr*	Your father তহতঁৰ দেউতাৰ *tahator deutAr*
3^{rd} তৃতীয়	Her father তাইৰ দেউতাক *tAir deutAk*	Their father সিহঁতৰ দেউতাক *sihator deutAk*

noun or personal pronoun they are considered particle. For example,

এই কাৰণে মই নগলোঁ।
TF[1] : *ei kArane moi nagalo.*
This + why + I+ did not go.
ET[2] : This is why I did not go.
ৰামৰ কাৰণে মই নগলোঁ।
TF : *rAmar kArane moi nagalo.*
Ram's + because of + I + did not go
ET : I did not go because of Ram.
In the first sentence কাৰণে (*kArne*) is placed after pronominal adjective এই (*ei*); so *kArne* is considered conjunction. But in the second sentence *kArne* is placed after noun ৰাম (*RAm*), and hence *kArne* is considered particle.

- Some prepositions or particles are used as suffix if they occur after noun, personal pronoun or verb. For example,

  সিহে গৈছিল। TF: *sihe goisil.*
  ET : Only he went.
  Actually হে (*he* : only) is a particle, but it is merged with the personal pronoun সি (*si*).

- An affix denoting number, gender or person, can be added to an adjective or other category word to create a noun word. For example,

  ধুনীয়াজনী হৈ আহিছা।
  TF : *dhuniyAjoni hoi aAhisA.*
  ET : You are looking beautiful.
  Here ধুনীয়া (*dhuniyA : beautiful*) is an adjective, but after adding feminine suffix জনী the whole constituent becomes a noun word.

- Even conjunctions can be used as other part of speech.
  হৰি আৰু যদু ভায়েক ককায়েক।
  TF : *Hari aAru Jadu bhAyek kokAyek.*
  ET : *Hari and Jadu are brothers.*
  যোৱাকালি ৰাতিৰ ঘটনাটোৱে বিষয়টোক আৰু অধিক ৰহস্যজনক কৰি তুলিলে।
  TF : *JowAkAli rAtir ghotonAtowe bishoitok aAru adhik rahashyajanak kori tulile.*
  ET : The last night incident has made the matter more mysterious.
  The word আৰু (*aAru : and*) shows ambiguity in these two sentences. In the first, it is used as conjunction (i.e. Hari and Jadu) and in the second, it is used as adjective of adjective.

3 Related Work

Several approaches have been used for building POS taggers. Two main approaches are supervised and unsupervised. Both supervised and unsupervised tagging can be of three sub-types. They are rule based, stochastic based and neural network based. There are number of pros and cons for each of these methods. The most common stochastic tagging technique is Hidden Markov Model (HMM).

During the last two decades, many different types of taggers have been developed, especially for corpus rich languages such as English. Nevertheless, due to relatively free word order, agglutinative nature, lack of resources and the general lateness in entering the computational linguistics field in India, reported tagger development work on Indian languages is relatively scanty. Among reported works, Dandapat (2007) developed a hybrid model of POS tagging by combining both supervised and unsupervised stochastic techniques. Avinesh and Karthik (2007) used conditional random field and transformation based learning. The heart of the system developed by Singh et al. (2006) for Hindi was the detailed linguistic analysis of morpho-syntactic phenomena, adroit handling of suffixes, accurate verb group identification and learning of disambiguation rules. Saha et al. (2004) developed a system for machine assisted POS tagging of Bangla corpora. Pammi and Prahllad (2007) developed a POS tagger and chunker using Decision Forests. This work explored different methods for POS tagging of Indian languages using sub-words as units. Generally, most POS taggers for Indian langauages use

[1]TF : Transliterated Assamese Form
[2]ET : Aproximate English Translation

morphological analyzer as a module. However, building morphological analyzer of a particular Indian language is a very difficult task.

4 Our Approach

We have used a Assamese text corpus (**Corpus Asm**) of nearly 300,000 words from the online version of the Assamese daily *Asomiya Pratidin* (Sharma et al., 2008). The downloaded articles use a font-based encoding called *Luit*. For our experiments we transliterate the texts to a normalised Roman encoding using transliteration software developed by us. We manually tag a part of this corpus, *Tr*, consisting of nearly 10,000 words for training. We use other portions of *Corpus Asm* for testing the tagger.

There was no tagset for Assamese before we started the project reported in this paper. Due to the morphological richness of the language, many words of Assamese occur in secondary forms in texts. This increases the number of POS tags that needed for the language. Also, often there are differences of opinion among linguists on the tags that may be associated with certain words in texts. We developed a tagset after in-depth consultation with linguists and manually tagged text segments of nearly 10,000 words according to their guidance. To make the tagging process easier we have subcategorised each category of noun and personal pronoun based on six case endings (viz, nominative, accussative, instumental, dative, genitive and locative) and two numbers.

We have used HMM (Dermatas and Kokkinakis, 1995) and the Viterbi algorithm (1967) in developing our POS tagger. HMM/Viterbi approach is the most useful method, when pretagged corpus is not available. First, in the training phase, we have manually tagged the *Tr* part of the corpus using the tagset discussed above. Then, we build four database tables using probabilities extracted from the manually tagged corpus- word-probability table, previous-tag-probability table, starting-tag-probability table and affix-probability table.

For testing, we consider three text segments, *A, B* and *C*, each of about 1000 words. First the input text is segmented into sentences. Each sentence is parsed individually. Each word of a sentence is stored in an array. After that, each word is searched in the word-probability table. If the word is unknown, its possible affixes are extracted and searched in the affix-probability table. From this search, we obtain the probable tags and their corresponding probabilities for each word. All these probable tags and the corresponding probabilities are stored in a two dimensional array which we call the *lattice* of the sentence. If we do not get probable tags and probabilities for a certain word from these two tables we assign tag CN (Common Noun) and probability 1 to the word since occurrence of CN is highest in the manually tagged corpus. After forming the lattice, the Viterbi algorithm is applied to the lattice that yields the most probable tag sequence for that sentence. After that next sentence is taken and the same procedure is repeated.

Table 2: POS tagging results with small corpora. Size of training words : 10000, UWH : Unknown word handling, UPH : Unknown proper noun handling

Test set	Size	Average accuracy	UDH accuracy	UPH accuracy
A	992	84.68%	62.8%	42.0%
B	1074	89.94%	67.54%	53.96%
C	1241	86.05%	85.64%	26.47%

Table 3: Comparison of our result with other HMM based model.

Author	Language	Average accuracy
Toutanova et al.(2003)	English	97.24%
Banko and Moore(2004)	English	96.55%
Dandapat and Sarkar(2006)	Bengali	84.37%
Rao et al.(2007)	Hindi	76.34%
	Bengali	72.17%
	Telegu	53.17%
Rao and Yarowsky(2007)	Hindi	70.67%
	Bengali	65.47%
	Telegu	65.85%
Sastry et al.(2007)	Hindi	69.98%
	Bengali	67.52%
	Telegu	68.32%
Ekbal et al.(2007)	Hindi	71.65%
	Bengali	80.63%
	Telegu	53.15%
Ours	Assamese	85.64%

5 Experimental Evaluation

The results using the three test segments are summarised in Table 2. The evaluation of the results require intensive manual verification effort. Larger training corpora is likely to produce more accurate results. More reliable results can be obtained using larger test corpora. Table 3 compares our result with other HMM based reported work. Form the table it is clear that

Toutanova et al. (2003) obtained the best result for English (97.24%). Among HMM based experiments reported on Indian languages, we have obtained the best result (86.89%). This work is ongoing and the corpus size and the amount of tagged text are being increased on a regular basis.

The accuracy of a tagger depends on the size of tagset used, vocabulary used, and size, genre and quality of the corpus used. Our tagset containing 172 tags is rather big compared to other Indian language tagsets. A smaller tagset is likely to give more accurate result, but may give less information about word structure and ambiguity. The corpora for training and testing our tagger are taken form an Assamese daily newspaper *Asomiya Pratidin*, thus they are of the same genre.

6 Conclusion & Future work

We have achieved good POS tagging results for Assamese, a fairly widely spoken language which had very little prior computational linguistic work. We have obtained an average tagging accuracy of 87% using a training corpus of just 10000 words. Our main achievement is the creation of the Assamese tagset that was not available before starting this project. We have implemented an existing method for POS tagging but our work is for a new language where an annotated corpora and a pre-defined tagset were not available.

We are currently working on developing a small and more compact tagset. We propose the following additional work for improved performance. First, the size of the manually tagged part of the corpus will have to be increased. Second, a suitable procedure for handling unknown proper nouns will have to be developed. Third, if this system can be expanded to trigrams or even n-grams using a larger training corpus, we believe that the tagging accuracy will increase.

Acknowledgemnt

We would like to thank Dr. Jyotiprakash Tamuli, Dr. Runima Chowdhary and Dr. Madhumita Barbora for their help, specially in making the Assamese tagset.

References

Avinesh PVS & Karthik G. POS tagging and chunking using Conditional Random Field and Transformation based learning. *IJCAI-07 workshop on Shallow Parsing for South Asian Languages*. 2007.

Banko, M., & Robert Moore, R. Part of speech tagging in context. *20th International Conference on Computational Linguistics*. 2004.

Dandapat, S. Part-of-Speech Tagging and Chunking with Maximum Entropy Model. *Workshop on Shallow Parsing for South Asian Languages*. 2007.

Dandapat, S., & Sarkar, S. Part-of-Speech Tagging for Bengali with Hidden Markov Model. *NLPAI ML workshop on Part of speech tagging and Chunking for Indian language*. 2006.

Dermatas, S., & Kokkinakis, G. Automatic stochastic tagging of natural language text. *Computational Linguistics* 21 : 137-163. 1995.

Ekbal, A., Mandal, S., & Bandyopadhyay, S. POS tagging using HMM and rule based chunking . *Workshop on Shallow Parsing for South Asian Languages*. 2007.

Goswami, G. C. Asamīyā Vyākaraṇ Pravesh, Second edition. *Bina Library, Guwahati*. 2003.

http://shiva.iiit.ac.in/SPSAL2007. *IJCAI-07 workshop on Shallow Parsing for South Asian Languages*. Hyderabad, India.

Pammi, S.C., & Prahallad, K. POS tagging and chunking using Decision Forests. *Workshop on Shallow Parsing for South Asian Languages*. 2007.

Rao, D., & Yarowsky, D.. Part of speech tagging and shallow parsing of Indian languages. *IJCAI-07 workshop on Shallow Parsing for South Asian Languages*. 2007.

Rao, P.T., & Ram, S.R., Vijaykrishna, R. & Sobha L. A text chunker and hybrid pos tagger for Indian languages. *IJCAI-07 workshop on Shallow Parsing for South Asian Languages*. 2007.

Saha, G.K., Saha, A.B., & Debnath, S. Computer Assisted Bangla Words POS Tagging. *Proc. International Symposium on Machine Translation NLP & TSS*. 2004.

Sastry, G.M.R., Chaudhuri, S., & Reddy, P.N. A HMM based part-of-speech and statistical chunker for 3 Indian languages. *IJCAI-07 workshop on Shallow Parsing for South Asian Languages*. 2007.

Sharma, U., Kalita, J. & Das, R. K. Acquisition of Morphology of an Indic language from text corpus. *ACM TALIP* 2008.

Singh, S., Gupta K., Shrivastava, M., & Bhattacharyya, P. Morphological richness offsets resource demand-experiences in constructing a POS tagger for Hindi. *COLING/ACL*. 2006.

Toutanova, K., Klein, D., Manning, C.D. & Singer, Y. Feature-Rich part-of-speech tagging with a Cyclic Dependency Network. *HLT-NAACL*. 2003.

Viterbi, A.J. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transaction on Information Theory* 61(3) : 268-278. 1967.

Improving data-driven dependency parsing using large-scale LFG grammars

Lilja Øvrelid, Jonas Kuhn and Kathrin Spreyer
Department of Linguistics
University of Potsdam
{lilja,kuhn,spreyer}@ling.uni-potsdam.de

Abstract

This paper presents experiments which combine a grammar-driven and a data-driven parser. We show how the conversion of LFG output to dependency representation allows for a technique of parser stacking, whereby the output of the grammar-driven parser supplies features for a data-driven dependency parser. We evaluate on English and German and show significant improvements stemming from the proposed dependency structure as well as various other, deep linguistic features derived from the respective grammars.

1 Introduction

The divide between grammar-driven and data-driven approaches to parsing has become less pronounced in recent years due to extensive work on robustness and efficiency for the grammar-driven approaches (Riezler et al., 2002; Cahill et al., 2008b). The linguistic generalizations captured in such knowledge-based resources are thus increasingly available for use in practical applications.

The NLP-community has in recent years witnessed a surge of interest in dependency-based approaches to syntactic parsing, spurred by the CoNLL shared tasks of dependency parsing (Buchholz and Marsi, 2006; Nivre et al., 2007). Nivre and McDonald (2008) show how two different approaches to dependency parsing, the graph-based and transition-based approaches, may be combined and subsequently learn to complement each other to achieve improved parse results for a range of different languages.

In this paper, we show how a data-driven dependency parser may straightforwardly be modified to learn directly from a grammar-driven parser. We evaluate on English and German and show significant improvements for both languages. Like Nivre and McDonald (2008), we supply a data-driven dependency parser with features from a different parser to guide parsing. The additional parser employed in this work, is not however, a data-driven parser trained on the same data set, but a grammar-driven parser outputing a deep LFG analysis. We furthermore show how a range of other features – morphological, structural and semantic – from the grammar-driven analysis may be employed during data-driven parsing and lead to significant improvements.

2 Grammar-driven LFG-parsing

The XLE system (Crouch et al., 2007) performs unification-based parsing using hand-crafted LFG grammars. It processes raw text and assigns to it both a phrase-structural ('c-structure') and a feature structural, functional ('f-structure').

In the work described in this paper, we employ the XLE platform using the grammars available for English and German from the ParGram project (Butt et al., 2002). In order to increase the coverage of the grammars, we employ the robustness techniques of fragment parsing and 'skimming' available in XLE (Riezler et al., 2002).

3 Dependency conversion and feature extraction

In extracting information from the output of the deep grammars we wish to capture as much of the precise, linguistic generalizations embodied in the grammars as possible, whilst keeping with the requirements posed by the dependency parser. The process is illustrated in Figure 1.

3.1 Data

The English data set consists of the Wall Street Journal sections 2-24 of the Penn treebank (Marcus et al., 1993), converted to dependency format. The treebank data used for German is the Tiger

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 37–40,
Suntec, Singapore, 4 August 2009.

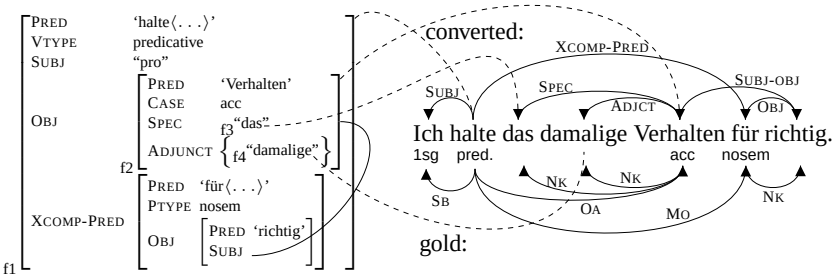


Figure 1: Treebank enrichment with LFG output; German example: *I consider the past behaviour correct.*

treebank (Brants et al., 2004), where we employ the version released with the CoNLL-X shared task on dependency parsing (Buchholz and Marsi, 2006).

3.2 LFG to dependency structure

We start out by converting the XLE output to a dependency representation. This is quite straightforward since the f-structures produced by LFG parsers can be interpreted as dependency structures. The conversion is performed by a set of rewrite rules which are executed by XLE's built-in extraction engine. We employ two strategies for the extraction of dependency structures from output containing multiple heads. We attach the dependent to the closest head and, i) label it with the corresponding label (Single), ii) label it with the complex label corresponding to the concatenation of the labels from the multiple head attachments (Complex). The converted dependency analysis in Figure 1 shows the f-structure and the corresponding converted dependency output of a German example sentence, where a raised object *Verhalten* receives the complex SUBJ-OBJ label. Following the XLE-parsing of the treebanks and the ensuing dependency conversion, we have a grammar-based analysis for 95.2% of the English sentence, 45238 sentences altogether, and 96.5% of the German sentences, 38189 sentences altogether.

3.3 Deep linguistic features

The LFG grammars capture linguistic generalizations which may not be reduced to a dependency representation. For instance, the grammars contain information on morphosyntactic properties such as case, gender and tense, as well as more semantic properties detailing various types of adverbials, specifying semantic conceptual categories such as human, time and location etc., see Figure 1. Table 1 presents the features extracted for use during parsing from the German and English XLE-parses.

4 Data-driven dependency parsing

MaltParser (Nivre et al., 2006a) is a language-independent system for data-driven dependency parsing which is freely available.[1] MaltParser is based on a deterministic parsing strategy in combination with treebank-induced classifiers for predicting parse transitions. MaltParser constructs parsing as a set of transitions between parse configurations. A parse configuration is a triple $\langle S, I, G \rangle$, where S represents the parse stack, I is the queue of remaining input tokens, and G represents the dependency graph defined thus far.

The feature model in MaltParser defines the relevant attributes of tokens in a parse configuration. Parse configurations are represented by a set of features, which focus on attributes of the *top* of the stack, the *next* input token and neighboring tokens in the stack, input queue and dependency graph under construction. Table 2 shows an example of a feature model.[2]

For the training of baseline parsers we employ feature models which make use of the word form (FORM), part-of-speech (POS) and the dependency relation (DEP) of a given token, exemplified in Table 2. For the baseline parsers and all subsequent parsers we employ the arg-eager algorithm in combination with SVM learners with a polynomial kernel.[3]

[1] http://maltparser.org

[2] Note that the feature model in Table 2 is an example feature model and not the actual model employed in the parse experiments. The details or references for the English and German models are provided below.

[3] For training of the baseline parsers we also employ some language-specific settings. For English we use learner and parser settings, as well as feature model from the English pretrained MaltParser-model available from http://maltparser.org. For German, we use the learner and parser settings from the parser employed in the CoNLL-X

POS	XFeats
Verb	CLAUSETYPE, GOVPREP, MOOD, PASSIVE, PERF, TENSE, VTYPE
Noun	CASE, COMMON, GOVPREP, LOCATIONTYPE, NUM, NTYPE, PERS, PROPERTYPE
Pronoun	CASE, GOVPREP, NUM, NTYPE, PERS
Prep	PSEM, PTYPE
Conj	COORD, COORD-FORM, COORD-LEVEL
Adv	ADJUNCTTYPE, ADVTYPE
Adj	ATYPE, DEGREE
English	DEVERBAL, PROG, SUBCAT, GENDSEM, HUMAN, TIME
German	AUXSELECT, AUXFLIP, COHERENT, FUT, DEF, GEND, GENITIVE, COUNT

Table 1: Features from XLE output, common for both languages and language-speciffic

	FORM	POS	DEP	XFEATS	XDEP
S:*top*	+	+	+	+	+
I:*next*	+	+		+	+
I:*next*-1	+			+	
G:head of *top*	+			+	
G:leftmost dependent of *top*			+	+	
InputArc(XHEAD)					

Table 2: Example feature model; S: stack, I: input, G: graph; $\pm n$ = n positions to the left($-$) or right ($+$).

5 Parser stacking

The procedure to enable the data-driven parser to learn from the grammar-driven parser is quite simple. We parse a treebank with the XLE platform. We then convert the LFG output to dependency structures, so that we have two parallel versions of the treebank – one gold standard and one with LFG-annotation. We extend the gold standard treebank with additional information from the corresponding LFG analysis, as illustrated by Figure 1 and train the data-driven dependency parser on the enhanced data set.

We extend the feature model of the baseline parsers in the same way as Nivre and McDonald (2008). The example feature model in Table 2 shows how we add the proposed dependency relation (XDEP) *top* and *next* as features for the parser. We furthermore add a feature which looks at whether there is an arc between these two tokens in the dependency structure (InputArc(XHEAD)), with three possible values: Left, Right, None. In order to incorporate further information supplied by the LFG grammars we extend the feature models with an additional, static attribute, XFEATS. This is employed for the range of deep linguistic features, detailed in section 3.3 above.

5.1 Experimental setup

All parse experiments are performed using 10-fold cross-validation for training and testing. Overall parsing accuracy will be reported using the standard metrics of labeled attachment score (LAS) and unlabeled attachment score (UAS).Statistical significance is checked using Dan Bikel's randomized parsing evaluation comparator.[4]

shared task (Nivre et al., 2006b). For both languages, we employ so-called "relaxed" root handling.

[4]http://www.cis.upenn.edu/∼dbikel/software.html

6 Results

We experiment with the addition of two types of features: i) the dependency structure proposed by XLE for a given sentence ii) other morphosyntactic, structural or lexical semantic features provided by the XLE grammar. The results are presented in Table 3.

For English, we find that the addition of proposed dependency structure from the grammar-driven parser causes a small, but significant improvement of results ($p<.0001$). In terms of labeled accuracy the results improve with 0.15 percentage points, from 89.64 to 89.79. The introduction of complex dependency labels to account for multiple heads in the LFG output causes a smaller improvement of results than the single labeling scheme. The corresponding results for German are presented in Table 3. We find that the addition of grammar-driven dependency structures with single labels (Single) improves the parse results significantly ($p<.0001$), both in terms of unlabeled and labeled accuracy. For labeled accuracy we observe an improvement of 1.45 percentage points, from 85.97 to 87.42. For the German data, we find that the addition of dependency structure with complex labels (Complex) gives a further small, but significant ($p<.03$) improvement over the experiment with single labels.

The results following the addition of the grammar-extracted features in Table 1 (Feats) are presented in Table 3.[5] We observe significant improvements of overall parse results for both languages ($p<.0001$).

[5]We experimented with several feature models for the inclusion of the additional information, however, found no significant differences when performing a forward feature selection. The simple feature model simply adds the XFEATS of the *top* and *next* tokens of the parse configuration.

	English		German	
	UAS	LAS	UAS	LAS
Baseline	92.48	89.64	88.68	85.97
Single	**92.61**	**89.79**	89.72	87.42
Complex	92.58	89.74	89.76	87.46
Feats	92.55	89.77	89.63	87.30
Single+Feats	92.52	89.69	90.01	87.77
Complex+Feats	92.53	89.70	**90.02**	**87.78**

Table 3: Overall results in experiments expressed as unlabeled and labeled attachment scores.

We also investigated combinations of the different sources of information – dependency structures and deep features. These results are presented in the final lines of Table 3. We find that for the English parser, the combination of the features do not cause a further improvement of results, compared to the individual experiments. The combined experiments (Single+Feats, Complex+Feats) for German, on the other hand, differ significantly from the baseline experiment, as well as the individual experiments (Single,Complex,Feats) reported above ($p<.0001$). By combination of the grammar-derived features we improve on the baseline by 1.81 percentage points.

A comparison with the German results obtained using MaltParser with graph-based dependency structures supplied by MSTParser (Nivre and McDonald, 2008) shows that our results using a grammar-driven parser largely corroborate the tendencies observed there. Our best results for German, combining dependency structures and additional features, slightly improve on those reported for MaltParser (by 0.11 percentage points).[6]

7 Conclusions and future work

This paper has presented experiments in the combination of a grammar-driven LFG-parser and a data-driven dependency parser. We have shown how the use of converted dependency structures in the training of a data-driven dependency parser, MaltParser, causes significant improvements in overall parse results for English and German. We have furthermore presented a set of additional, deep features which may straightforwardly be extracted from the grammar-based output and cause individual improvements for both languages and a combined effect for German.

In terms of future work, a more extensive error analysis will be performed to locate the precise benefits of the parser combination. We will also investigate the application of the method directly to raw text and application to a task which may benefit specifically from the combined analyses, such as semantic role labeling or semantic verb classification.

It has recently been shown that automatically acquired LFG grammars may actually outperform hand-crafted grammars in parsing (Cahill et al., 2008a). These results add further to the relevance of the results shown in this paper, bypassing the bottleneck of grammar hand-crafting as a prerequisite for the applicability of our results.

[6] English was not among the languages investigated in Nivre and McDonald (2008).

References

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther Knig, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a German corpus. *Research on Language and Computation*, 2:597–620.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL-X)*.

Miriam Butt, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of COLING-2002 Workshop on Grammar Engineering and Evaluation.*

Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith, and Andy Way. 2008a. Wide-coverage deep statistical parsing using automatic dependency structure annotation. *Computational Linguistics*.

Aoife Cahill, John T. Maxwell, Paul Meurer, Christian Rohrer, and Victoria Rosen. 2008b. Speeding up LFG parsing using c-structure pruning. In *Proceedings of the Workshop on Grammar Engineering Across Frameworks*.

D. Crouch, M. Dalrymple, R. Kaplan, T. King, J. Maxwell, and P. Newman, 2007. *XLE Documentation*. http://www2.parc.com/isl/.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus for English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-HLT 2008*.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006a. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC*.

Joakim Nivre, Jens Nilsson, Johan Hall, Gülşen Eryiğit, and Svetoslav Marinov. 2006b. Labeled pseudo-projective dependency parsing with Support Vector Machines. In *Proceedings of CoNLL*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.

Stefan Riezler, Tracy King, Ronald Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. 2002. Parsing the Wall Street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of ACL*.

Incremental Parsing with Monotonic Adjoining Operation

Yoshihide Kato and **Shigeki Matsubara**
Information Technology Center, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601 Japan
{yosihide,matubara}@el.itc.nagoya-u.ac.jp

Abstract

This paper describes an incremental parser based on an adjoining operation. By using the operation, we can avoid the problem of infinite local ambiguity in incremental parsing. This paper further proposes a restricted version of the adjoining operation, which preserves lexical dependencies of partial parse trees. Our experimental results showed that the restriction enhances the accuracy of the incremental parsing.

1 Introduction

Incremental parser reads a sentence from left to right, and produces partial parse trees which span all words in each initial fragment of the sentence. Incremental parsing is useful to realize real-time spoken language processing systems, such as a simultaneous machine interpretation system, an automatic captioning system, or a spoken dialogue system (Allen et al., 2001).

Several incremental parsing methods have been proposed so far (Collins and Roark, 2004; Roark, 2001; Roark, 2004). In these methods, the parsers can produce the candidates of partial parse trees on a word-by-word basis. However, they suffer from the problem of infinite local ambiguity, i.e., they may produce an infinite number of candidates of partial parse trees. This problem is caused by the fact that partial parse trees can have arbitrarily nested left-recursive structures and there is no information to predict the depth of nesting.

To solve the problem, this paper proposes an incremental parsing method based on an adjoining operation. By using the operation, we can avoid the problem of infinite local ambiguity. This approach has been adopted by Lombardo and Sturt (1997) and Kato et al. (2004). However, this raises another problem that their adjoining operations cannot preserve lexical dependencies of partial parse trees. This paper proposes a restricted version of the adjoining operation which preserves lexical dependencies. Our experimental results showed that the restriction enhances the accuracy of the incremental parsing.

2 Incremental Parsing

This section gives a description of Collins and Roark's incremental parser (Collins and Roark, 2004) and discusses its problem.

Collins and Roark's parser uses a grammar defined by a 6-tuple $G = (V, T, S, \#, C, B)$. V is a set of nonterminal symbols. T is a set of terminal symbols. S is called a start symbol and $S \in V$. $\#$ is a special symbol to mark the end of a constituent. The rightmost child of every parent is labeled with this symbol. This is necessary to build a proper probabilistic parsing model. C is a set of *allowable chains*. An allowable chain is a sequence of nonterminal symbols followed by a terminal symbol. Each chain corresponds to a label sequence on a path from a node to its leftmost descendant leaf. B is a set of *allowable triples*. An allowable triple is a tuple $\langle X, Y, Z \rangle$ where $X, Y, Z \in V$. The triple specifies which nonterminal symbol Z is allowed to follow a nonterminal symbol Y under a parent X.

For each initial fragment of a sentence, Collins and Roark's incremental parser produces partial parse trees which span all words in the fragment.

Let us consider the parsing process as shown in Figure 1. For the first word "we", the parser produces the partial parse tree (a), if the allowable chain $\langle \text{S} \rightarrow \text{NP} \rightarrow \text{PRP} \rightarrow \text{we} \rangle$ exists in C. For other chains which start with S and end with "we", the parser produces partial parse trees by using the chains. For the next word, the parser attaches the chain $\langle \text{VP} \rightarrow \text{VBP} \rightarrow \text{describe} \rangle$ to the partial parse tree (a) [1]. The attachment is possible when the allowable triple $\langle$S, NP, VP$\rangle$ exists in B.

[1] More precisely, the chain is attached after attaching end-of-constituent # under the NP node.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 41–44,
Suntec, Singapore, 4 August 2009.

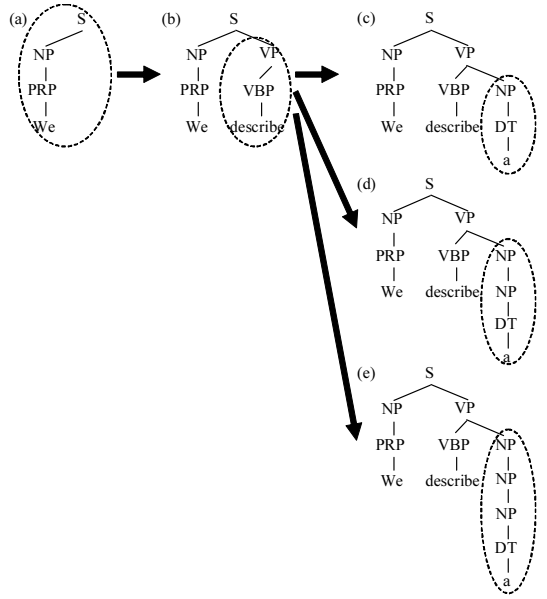


Figure 1: A process in incremental parsing

2.1 Infinite Local Ambiguity

Incremental parsing suffers from the problem of infinite local ambiguity. The ambiguity is caused by left-recursion. An infinite number of partial parse trees are produced, because we cannot predict the depth of left-recursive nesting.

Let us consider the fragment "We describe a." For this fragment, there exist several candidates of partial parse trees. Figure 1 shows candidates of partial parse trees. The partial parse tree (c) represents that the noun phrase which starts with "a" has no adjunct. The tree (d) represents that the noun phrase has an adjunct or is a conjunct of a coordinated noun phrase. The tree (e) represents that the noun phrase has an adjunct and the noun phrase with an adjunct is a conjunct of a coordinated noun phrase. The partial parse trees (d) and (e) are the instances of partial parse trees which have left-recursive structures. The major problem is that there is no information to determine the depth of left-recursive nesting at this point.

3 Incremental Parsing Method Based on Adjoining Operation

In order to avoid the problem of infinite local ambiguity, the previous works have adopted the following approaches: (1) a beam search strategy (Collins and Roark, 2004; Roark, 2001; Roark, 2004), (2) limiting the allowable chains to those actually observed in the treebank (Collins and Roark, 2004), and (3) transforming the parse trees with a selective left-corner transformation (Johnson and Roark, 2000) before inducing the allowable chains and allowable triples (Collins and Roark, 2004). The first and second approaches can prevent the parser from infinitely producing partial parse trees, but the parser has to produce partial parse trees as shown in Figure 1. The local ambiguity still remains. In the third approach, no left recursive structure exists in the transformed grammar, but the parse trees defined by the grammar are different from those defined by the original grammar. It is not clear if partial parse trees defined by the transformed grammar represent syntactic relations correctly.

As an approach to solve these problems, we introduce an adjoining operation to incremental parsing. Lombardo and Sturt (1997) and Kato et al. (2004) have already adopted this approach. However, their methods have another problem that their adjoining operations cannot preserve lexical dependencies of partial parse trees. To solve this problem, this section proposes a restricted version of the adjoining operation.

3.1 Adjoining Operation

An adjoining operation is used in Tree-Adjoining Grammar (Joshi, 1985). The operation inserts a tree into another tree. The inserted tree is called an *auxiliary tree*. Each auxiliary tree has a leaf called a *foot* which has the same nonterminal symbol as its root. An adjoining operation is defined as follows:

adjoining An adjoining operation splits a parse tree σ at a nonterminal node η and inserts an auxiliary tree β having the same nonterminal symbol as η, i.e., combines the upper tree of σ with the root of β and the lower tree of σ with the foot of β.

We write $a_{\eta,\beta}(\sigma)$ for the partial parse tree obtained by adjoining β to σ at η.

We use simplest auxiliary trees, which consist of a root and a foot.

As we have seen in Figure 1, Collins and Roark's parser produces partial parse trees such as (c), (d) and (e). On the other hand, by using the adjoining operation, our parser produces only the partial parse tree (c). When a left-recursive structure is required to parse the sentence, our parser adjoins it. In the example above, the parser adjoins the auxiliary tree $\langle \text{NP} \rightarrow \text{NP} \rangle$ to the partial parse tree (c) when the word "for" is read. This enables

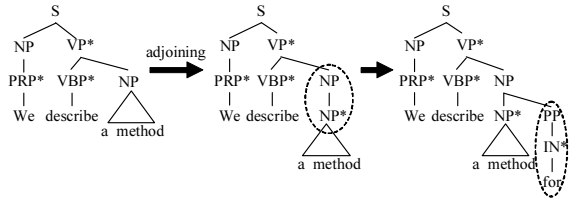


Figure 2: Adjoining operation

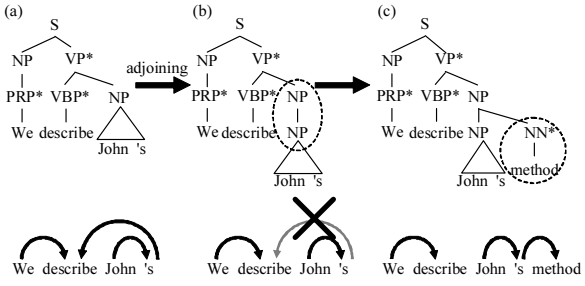


Figure 3: Non-monotonic adjoining operation

the parser to attach the allowable chain ⟨PP → IN → for⟩. The parsing process is shown in Figure 2.

3.2 Adjoining Operation and Monotonicity

By using the adjoining operation, we avoid the problem of infinite local ambiguity. However, the adjoining operation cannot preserve lexical dependencies of partial parse trees. Lexical dependency is a kind of relation between words, which represents head-modifier relation. We can map parse trees to sets of lexical dependencies by identifying the head-child of each constituent in the parse tree (Collins, 1999).

Let us consider the parsing process as shown in Figure 3. The partial parse tree (a) is a candidate for the initial fragment "We describe John 's". We mark each head-child with a special symbol $*$. We obtain three lexical dependencies ⟨We → describe⟩, ⟨John → 's⟩ and ⟨'s → describe⟩ from (a). When the parser reads the next word "method", it produces the partial parse tree (b) by adjoining the auxiliary tree ⟨NP → NP⟩. The partial parse tree (b) does not have ⟨'s → describe⟩. The dependency ⟨'s → describe⟩ is removed when the parser adjoins the auxiliary tree ⟨NP → NP⟩ to (a). This example demonstrates that the adjoining operation cannot preserve lexical dependencies of partial parse trees.

Now, we define the monotonicity of the adjoining operation. We say that adjoining an auxiliary tree β to a partial parse tree σ at a node η is *monotonic* when $dep(\sigma) \subseteq dep(a_{\eta,\beta}(\sigma))$ where dep is the mapping from a parse tree to a set of dependencies. An auxiliary tree β is monotonic if adjoining β to any partial parse tree is monotonic.

We want to exclude any non-monotonic auxiliary tree from the grammar. For this purpose, we restrict the form of auxiliary trees. In our framework, all auxiliary trees satisfy the following constraint:

- The foot of each auxiliary tree must be the head-child of its parent.

The auxiliary tree ⟨NP → NP*⟩ satisfies the constraint, while ⟨NP → NP⟩ does not.

3.3 Our Incremental Parser

Our incremental parser is based on a probabilistic parsing model which assigns a probability to each operation. The probability of a partial parse tree is defined by the product of the probabilities of the operations used in its construction. The probability of attaching an allowable chain c to a partial parse tree σ is approximated as follows:

$$\begin{aligned} P(c \mid \sigma) \quad = \quad & P_{root}(R \mid \mathbf{P}, \mathbf{L}, H, t_H, w_H, \mathbf{D}) \\ & \times P_{template}(c' \mid R, \mathbf{P}, \mathbf{L}, H) \\ & \times P_{word}(w \mid c', t_h, w_h) \end{aligned}$$

where R is the root label of c, c' is the sequence which is obtained by omitting the last element from c and w is the last element of c. The probability is conditioned on a limited context of σ. $\mathbf{P}$ is a set of the ancestor labels of R. $\mathbf{L}$ is a set of the left-sibling labels of R. H is the head label in $\mathbf{L}$. w_H and t_H are the head word and head tag of H, respectively. $\mathbf{D}$ is a set of distance features. w_h and t_h are the word and POS tag modified by w, respectively. The adjoining probability is approximated as follows:

$$P(\beta \mid \sigma) = P_{adjoining}(\beta \mid \mathbf{P}, \mathbf{L}, H, \mathbf{D})$$

where β is an auxiliary tree or a special symbol nil, the nil means that no auxiliary tree is adjoined. The limited contexts used in this model are similar to the previous methods (Collins and Roark, 2004; Roark, 2001; Roark, 2004).

To achieve efficient parsing, we use a beam search strategy like the previous methods (Collins and Roark, 2004; Roark, 2001; Roark, 2004). For each word position i, our parser has a priority queue H_i. Each queue H_i stores the only N-best

Table 1: Parsing results

	LR(%)	LP(%)	F(%)
Roark (2004)	86.4	86.8	86.6
Collins and Roark (2004)	86.5	86.8	86.7
No adjoining	86.3	86.8	86.6
Non-monotonic adjoining	86.1	87.1	86.6
Monotonic adjoining	87.2	87.7	87.4

partial parse trees. In addition, the parser discards the partial parse tree σ whose probability $P(\sigma)$ is less than the $P^*\gamma$ where P^* is the highest probability on the queue H_i and γ is a beam factor.

4 Experimental Evaluation

To evaluate the performance of our incremental parser, we conducted a parsing experiment. We implemented the following three types of incremental parsers to assess the influence of the adjoining operation and its monotonicity: (1) without adjoining operation, (2) with non-monotonic adjoining operation, and (3) with monotonic adjoining operation. The grammars were extracted from the parse trees in sections 02-21 of the Wall Street Journal in Penn Treebank. We identified the head-child in each constituent by using the head rule of Collins (Collins, 1999). The probabilistic models were built by using the maximum entropy method. We set the beam-width N to 300 and the beam factor γ to 10^{-11}.

We evaluated the parsing accuracy by using section 23. We measured labeled recall and labeled precision. Table 1 shows the results[2]. Our incremental parser is competitive with the previous ones. The incremental parser with the monotonic adjoining operation outperforms the others. The result means that our proposed constraint of auxiliary trees improves parsing accuracy.

5 Conclusion

This paper has proposed an incremental parser based on an adjoining operation to solve the problem of infinite local ambiguity. The adjoining operation causes another problem that the parser cannot preserve lexical dependencies of partial parse trees. To tackle this problem, we defined the monotonicity of adjoining operation and restricted the form of auxiliary trees to satisfy the constraint of the monotonicity. Our experimental result showed that the restriction improved the accuracy of our incremental parser.

In future work, we will investigate the incremental parser for head-final language such as Japanese. Head-final language includes many indirect left-recursive structures. In this paper, we dealt with direct left-recursive structures only. To process indirect left-recursive structures, we need to extend our method.

[2]The best results of Collins and Roark (2004) (LR=88.4%, LP=89.1% and F=88.8%) are achieved when the parser utilizes the information about the final punctuation and the look-ahead. However, the parsing process is not on a word-by-word basis. The results shown in Table 1 are achieved when the parser does not utilize such informations.

References

James Allen, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In *Proceedings of International Conference of Intelligent User Interfaces*, pages 1–8.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 111–118, Barcelona, Spain, July.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Mark Johnson and Brian Roark. 2000. Compact non-left-recursive grammars using the selective left-corner transform and factoring. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 355–361, July.

Aravind K. Joshi. 1985. Tree adjoining grammars: How much context sensitivity is required to provide a reasonable structural description? In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press.

Yoshihide Kato, Shigeki Matsubara, and Yasuyoshi Inagaki. 2004. Stochastically evaluating the validity of partial parse trees in incremental parsing. In *Proceedings of the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 9–15, July.

Vincenzo Lombardo and Patrick Sturt. 1997. Incremental processing and infinite local ambiguity. In *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, pages 448–453.

Brian Roark. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249–276, June.

Brian Roark. 2004. Robust garden path parsing. *Natural language engineering*, 10(1):1–24.

Bayesian Learning of a Tree Substitution Grammar

Matt Post and **Daniel Gildea**
Department of Computer Science
University of Rochester
Rochester, NY 14627

Abstract

Tree substitution grammars (TSGs) offer many advantages over context-free grammars (CFGs), but are hard to learn. Past approaches have resorted to heuristics. In this paper, we learn a TSG using Gibbs sampling with a nonparametric prior to control subtree size. The learned grammars perform significantly better than heuristically extracted ones on parsing accuracy.

1 Introduction

Tree substition grammars (TSGs) have potential advantages over regular context-free grammars (CFGs), but there is no obvious way to learn these grammars. In particular, learning procedures are not able to take direct advantage of manually annotated corpora like the Penn Treebank, which are not marked for derivations and thus assume a standard CFG. Since different TSG derivations can produce the same parse tree, learning procedures must guess the derivations, the number of which is exponential in the tree size. This compels heuristic methods of subtree extraction, or maximum likelihood estimators which tend to extract large subtrees that overfit the training data.

These problems are common in natural language processing tasks that search for a hidden segmentation. Recently, many groups have had success using Gibbs sampling to address the complexity issue and nonparametric priors to address the overfitting problem (DeNero et al., 2008; Goldwater et al., 2009). In this paper we apply these techniques to learn a tree substitution grammar, evaluate it on the Wall Street Journal parsing task, and compare it to previous work.

2 Model

2.1 Tree substitution grammars

TSGs extend CFGs (and their probabilistic counterparts, which concern us here) by allowing nonterminals to be rewritten as subtrees of arbitrary size. Although nonterminal rewrites are still context-free, in practice TSGs can loosen the independence assumptions of CFGs because larger rules capture more context. This is simpler than the complex independence and backoff decisions of Markovized grammars. Furthermore, subtrees with terminal symbols can be viewed as learning dependencies among the words in the subtree, obviating the need for the manual specification (Magerman, 1995) or automatic inference (Chiang and Bikel, 2002) of lexical dependencies.

Following standard notation for PCFGs, the probability of a derivation d in the grammar is given as

$$\Pr(d) = \prod_{r \in d} \Pr(r)$$

where each r is a rule used in the derivation. Under a regular CFG, each parse tree uniquely idenfifies a derivation. In contrast, multiple derivations in a TSG can produce the same parse; obtaining the parse probability requires a summation over all derivations that could have produced it. This disconnect between parses and derivations complicates both inference and learning. The inference (parsing) task for TSGs is NP-hard (Sima'an, 1996), and in practice the most probable parse is approximated (1) by sampling from the derivation forest or (2) from the top k derivations.

Grammar learning is more difficult as well. CFGs are usually trained on treebanks, especially the Wall Street Journal (WSJ) portion of the Penn Treebank. Once the model is defined, relevant

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 45–48,
Suntec, Singapore, 4 August 2009.

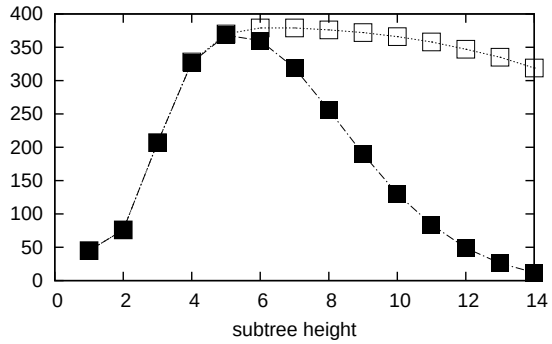


Figure 1: Subtree count (thousands) across heights for the "all subtrees" grammar (□) and the superior "minimal subset" (■) from Bod (2001).

events can simply be counted in the training data. In contrast, there are no treebanks annotated with TSG derivations, and a treebank parse tree of n nodes is ambiguous among 2^n possible derivations. One solution would be to manually annotate a treebank with TSG derivations, but in addition to being expensive, this task requires one to know what the grammar actually is. Part of the thinking motivating TSGs is to let the data determine the best set of subtrees.

One approach to grammar-learning is Data-Oriented Parsing (DOP), whose strategy is to simply take *all* subtrees in the training data as the grammar (Bod, 1993). Bod (2001) did this, approximating "all subtrees" by extracting from the Treebank 400K random subtrees for each subtree height ranging from two to fourteen, and compared the performance of that grammar to that of a heuristically pruned "minimal subset" of it. The latter's performance was quite good, achieving 90.8% F_1 score[1] on section 23 of the WSJ.

This approach is unsatisfying in some ways, however. Instead of heuristic extraction we would prefer a model that explained the subtrees found in the grammar. Furthermore, it seems unlikely that subtrees with ten or so lexical items will be useful on average at test time (Bod did not report how often larger trees are used, but did report that including subtrees with up to twelve lexical items improved parser performance). We expect there to be fewer large subtrees than small ones. Repeating Bod's grammar extraction experiment, this is indeed what we find when comparing these two grammars (Figure 1).

In summary, we would like a principled (model-based) means of determining from the data which set of subtrees should be added to our grammar, and we would like to do so in a manner that prefers smaller subtrees but permits larger ones if the data warrants it. This type of requirement is common in NLP tasks that require searching for a hidden segmentation, and in the following sections we apply it to learning a TSG from the Penn Treebank.

2.2 Collapsed Gibbs sampling with a DP prior[2]

For an excellent introduction to collapsed Gibbs sampling with a DP prior, we refer the reader to Appendix A of Goldwater et al. (2009), which we follow closely here. Our training data is a set of parse trees $\mathcal{T}$ that we assume was produced by an unknown TSG g with probability $\Pr(\mathcal{T}|g)$. Using Bayes' rule, we can compute the probability of a particular hypothesized grammar as

$$\Pr(g \mid \mathcal{T}) = \frac{\Pr(\mathcal{T} \mid g)\Pr(g)}{\Pr(\mathcal{T})}$$

$\Pr(g)$ is a distribution over grammars that expresses our *a priori* preference for g. We use a set of Dirichlet Process (DP) priors (Ferguson, 1973), one for each nonterminal $X \in N$, the set of nonterminals in the grammar. A sample from a DP is a distribution over events in an infinite sample space (in our case, potential subtrees in a TSG) which takes two parameters, a base measure and a concentration parameter:

$$\begin{aligned} g_X &\sim DP(G_X, \alpha) \\ G_X(t) &= \Pr_\$(|t|; p_\$) \prod_{r \in t} \Pr_{\text{MLE}}(r) \end{aligned}$$

The base measure G_X defines the probability of a subtree t as the product of the PCFG rules $r \in t$ that constitute it and a geometric distribution $\Pr_\$$ over the number of those rules, thus encoding a preference for smaller subtrees.[3] The parameter α contributes to the probability that previously unseen subtrees will be sampled. All DPs share parameters $p_\$$ and α. An entire grammar is then given as $g = \{g_X : X \in N\}$. We emphasize that no head information is used by the sampler.

Rather than explicitly consider each segmentation of the parse trees (which would define a TSG and its associated parameters), we use a collapsed Gibbs sampler to integrate over all possi-

[1] The harmonic mean of precision and recall: $F_1 = \frac{2PR}{P+R}$.

[2] Cohn et al. (2009) and O'Donnell et al. (2009) independently developed similar models.

[3] $G_X(t) = 0$ unless root$(t) = X$.

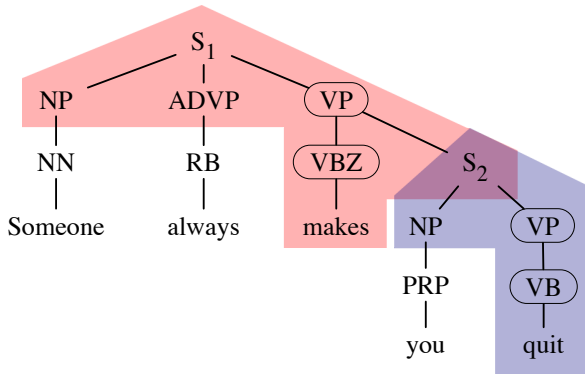


Figure 2: Depiction of $\overline{\text{sub}}(S_2)$ and $\underline{\text{sub}}(S_2)$. Highlighted subtrees correspond with our spinal extraction heuristic (§3). Circles denote nodes whose flag=1.

ble grammars and sample directly from the posterior. This is based on the Chinese Restaurant Process (CRP) representation of the DP. The Gibbs sampler is an iterative procedure. At initialization, each parse tree in the corpus is annotated with a specific derivation by marking each node in the tree with a binary flag. This flag indicates whether the subtree rooted at that node (a height one CFG rule, at minimum) is part of the subtree containing its parent. The Gibbs sampler considers every non-terminal, non-root node c of each parse tree in turn, freezing the rest of the training data and randomly choosing whether to join the subtrees above c and rooted at c (outcome h_1) or to split them (outcome h_2) according to the probability ratio $\phi(h_1)/(\phi(h_1)+\phi(h_2))$, where ϕ assigns a probability to each of the outcomes (Figure 2).

Let $\overline{\text{sub}}(n)$ denote the subtree above and including node n and $\underline{\text{sub}}(n)$ the subtree rooted at n; $\circ$ is a binary operator that forms a single subtree from two adjacent ones. The outcome probabilities are:

$$\begin{aligned}\phi(h_1) &= \theta(t)\\ \phi(h_2) &= \theta(\overline{\text{sub}}(c)) \cdot \theta(\underline{\text{sub}}(c))\end{aligned}$$

where $t = \overline{\text{sub}}(c) \circ \underline{\text{sub}}(c)$. Under the CRP, the subtree probability $\theta(t)$ is a function of the current state of the rest of the training corpus, the appropriate base measure $G_{\text{root}(t)}$, and the concentration parameter α:

$$\theta(t) = \frac{\text{count}_{z_t}(t) + \alpha G_{\text{root}(t)}(t)}{|z_t| + \alpha}$$

where z_t is the multiset of subtrees in the frozen portion of the training corpus sharing the same root as t, and $\text{count}_{z_t}(t)$ is the count of subtree t among them.

3 Experiments

3.1 Setup

We used the standard split for the Wall Street Journal portion of the Treebank, training on sections 2 to 21, and reporting results on sentences with no more than forty words from section 23.

We compare with three other grammars.

- A standard Treebank PCFG.
- A "spinal" TSG, produced by extracting n lexicalized subtrees from each length n sentence in the training data. Each subtree is defined as the sequence of CFG rules from leaf upward all sharing a head, according to the Magerman head-selection rules. We detach the top-level unary rule, and add in counts from the Treebank CFG rules.
- An in-house version of the heuristic "minimal subset" grammar of Bod (2001).[4]

We note two differences in our work that explain the large difference in scores for the minimal grammar from those reported by Bod: (1) we did not implement the smoothed "mismatch parsing", which permits lexical leaves of subtrees to act as wildcards, and (2) we approximate the most probable parse with the top single derivation instead of the top 1,000.

Rule probabilities for all grammars were set with relative frequency. The Gibbs sampler was initialized with the spinal grammar derivations. We construct sampled grammars in two ways: by summing all subtree counts from the derivation states of the first i sampling iterations together with counts from the Treebank CFG rules (denoted $(\alpha, p_\$, \leq i)$), and by taking the counts only from iteration i (denoted $(\alpha, p_\$, i)$).

Our standard CKY parser and Gibbs sampler were both written in Perl. TSG subtrees were flattened to CFG rules and reconstructed afterward, with identical mappings favoring the most probable rule. For pruning, we binned nonterminals according to input span and degree of binarization, keeping the ten highest scoring items in each bin.

3.2 Results

Table 1 contains parser scores. The spinal TSG outperforms a standard unlexicalized PCFG and

[4] All rules of height one, plus 400K subtrees sampled at each height $h, 2 \leq h \leq 14$, minus unlexicalized subtrees of $h > 6$ and lexicalized subtrees with more than twelve words.

grammar	size	LP	LR	F_1
PCFG	46K	75.37	70.05	72.61
spinal	190K	80.30	78.10	**79.18**
minimal subset	2.56M	76.40	78.29	77.33
$(10, 0.7, 100)$	62K	81.48	81.03	81.25
$(10, 0.8, 100)$	61K	81.23	80.79	81.00
$(10, 0.9, 100)$	61K	82.07	81.17	**81.61**
$(100, 0.7, 100)$	64K	81.23	80.98	81.10
$(100, 0.8, 100)$	63K	82.13	81.36	**81.74**
$(100, 0.9, 100)$	62K	82.11	81.20	81.65
$(100, 0.7, \leq 100)$	798K	82.38	82.27	82.32
$(100, 0.8, \leq 100)$	506K	82.27	81.95	82.10
$(100, 0.9, \leq 100)$	290K	82.64	82.09	**82.36**
$(100, 0.7, 500)$	61K	81.95	81.76	81.85
$(100, 0.8, 500)$	60K	82.73	82.21	**82.46**
$(100, 0.9, 500)$	59K	82.57	81.53	82.04
$(100, 0.7, \leq 500)$	2.05M	82.81	82.01	82.40
$(100, 0.8, \leq 500)$	1.13M	83.06	82.10	**82.57**
$(100, 0.9, \leq 500)$	528K	83.17	81.91	82.53

Table 1: Labeled precision, recall, and F_1 on WSJ§23.

the significantly larger "minimal subset" grammar. The sampled grammars outperform all of them. Nearly all of the rules of the best single iteration sampled grammar $(100, 0.8, 500)$ are lexicalized (50,820 of 60,633), and almost half of them have a height greater than one (27,328). Constructing sampled grammars by summing across iterations improved over this in all cases, but at the expense of a much larger grammar.

Figure 3 shows a histogram of subtree size taken from the counts of the subtrees (by token, not type) actually used in parsing WSJ§23. Parsing with the "minimal subset" grammar uses highly lexicalized subtrees, but they do not improve accuracy. We examined sentence-level F_1 scores and found that the use of larger subtrees did correlate with accuracy; however, the low overall accuracy (and the fact that there are so many of these large subtrees available in the grammar) suggests that such rules are overfit. In contrast, the histogram of subtree sizes used in parsing with the sampled grammar matches the shape of the histogram from the grammar itself. Gibbs sampling with a DP prior chooses smaller but more general rules.

4 Summary

Collapsed Gibbs sampling with a DP prior fits nicely with the task of learning a TSG. The sampled grammars are model-based, are simple to specify and extract, and take the expected shape

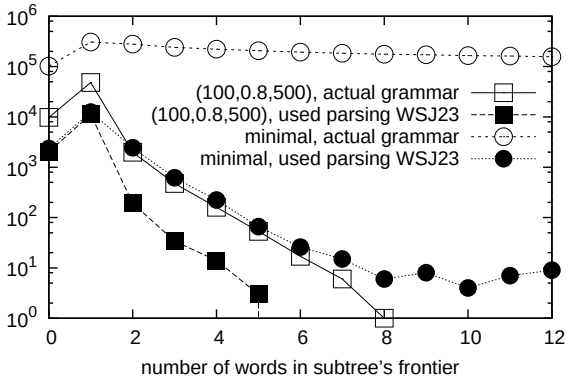


Figure 3: Histogram of subtrees sizes used in parsing WSJ§23 (filled points), as well as from the grammars themselves (outlined points).

over subtree size. They substantially outperform heuristically extracted grammars from previous work as well as our novel spinal grammar, and can do so with many fewer rules.

Acknowledgments This work was supported by NSF grants IIS-0546554 and ITR-0428020.

References

Rens Bod. 1993. Using an annotated corpus as a stochastic grammar. In *Proc. ACL*.

Rens Bod. 2001. What is the minimal set of fragments that achieves maximal parse accuracy. In *Proc. ACL*.

David Chiang and Daniel M. Bikel. 2002. Recovering latent information in treebanks. In *COLING*.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proc. NAACL*.

John DeNero, Alexandre Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *EMNLP*.

Thomas S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *Annals of Mathematical Statistics*, 1(2):209–230.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*.

David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Proc. ACL*.

T.J. O'Donnell, N.D. Goodman, J. Snedeker, and J.B. Tenenbaum. 2009. Computation and reuse in language. In *Proc. Cognitive Science Society*.

Khalil Sima'an. 1996. Computational complexity of probabilistic disambiguation by means of tree grammars. In *COLING*.

A Unified Single Scan Algorithm for Japanese Base Phrase Chunking and Dependency Parsing

Manabu Sassano
Yahoo Japan Corporation
Midtown Tower,
9-7-1 Akasaka, Minato-ku,
Tokyo 107-6211, Japan
msassano@yahoo-corp.jp

Sadao Kurohashi
Graduate School of Informatics,
Kyoto University
Yoshida-honmachi, Sakyo-ku,
Kyoto 606-8501, Japan
kuro@i.kyoto-u.ac.jp

Abstract

We describe an algorithm for Japanese analysis that does both base phrase chunking and dependency parsing simultaneously in linear-time with a single scan of a sentence. In this paper, we show a pseudo code of the algorithm and evaluate its performance empirically on the Kyoto University Corpus. Experimental results show that the proposed algorithm with the voted perceptron yields reasonably good accuracy.

1 Introduction

Single scan algorithms of parsing are important for interactive applications of NLP. For instance, such algorithms would be more suitable for robots accepting speech inputs or chatbots handling natural language inputs which should respond quickly in some situations even when human inputs are not clearly ended.

Japanese sentence analysis typically consists of three major steps, namely morphological analysis, *bunsetsu* (base phrase) chunking, and dependency parsing. In this paper, we describe a novel algorithm that combines the last two steps into a single scan process. The algorithm, which is an extension of Sassano's (2004), allows us to chunk morphemes into base phrases and decide dependency relations of the phrases in a strict left-to-right manner. We show a pseudo code of the algorithm and evaluate its performance empirically with the voted perceptron on the Kyoto University Corpus (Kurohashi and Nagao, 1998).

2 Japanese Sentence Structure

In Japanese NLP, it is often assumed that the structure of a sentence is given by dependency relations among *bunsetsu*s. A *bunsetsu* is a base phrasal unit and consists of one or more content words followed by zero or more function words.

	Meg-ga	kare-ni	ano	pen-wo	age-ta.
	Meg-subj	to him	that	pen-acc	give-past.
ID	0	1	2	3	4
Head	4	4	3	4	-

Figure 1: Sample sentence (bunsetsu-based)

In addition, most of algorithms of Japanese dependency parsing, e.g., (Sekine et al., 2000; Sassano, 2004), assume the three constraints below. (1) Each bunsetsu has only one head except the rightmost one. (2) Dependency links between bunsetsus go from left to right. (3) Dependency links do not cross one another. In other words, dependencies are projective.

A sample sentence in Japanese is shown in Figure 1. We can see all the constraints are satisfied.

3 Previous Work

As far as we know, there is no dependency parser that does simultaneously both bunsetsu chunking and dependency parsing and, in addition, does them with a single scan. Most of the modern dependency parsers for Japanese require *bunsetsu* chunking (base phrase chunking) before dependency parsing (Sekine et al., 2000; Kudo and Matsumoto, 2002; Sassano, 2004). Although word-based parsers are proposed in (Mori et al., 2000; Mori, 2002), they do not build bunsetsus and are not compatible with other Japanese dependency parsers. Multilingual parsers of participants in the CoNLL 2006 shared task (Buchholz and Marsi, 2006) can handle Japanese sentences. But they are basically word-based.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 49–52,
Suntec, Singapore, 4 August 2009.

	Meg	ga	kare	ni	ano	pen	wo	age-ta.
	Meg	subj	him	to	that	pen	acc	give-past.
ID	0	1	2	3	4	5	6	7
Head	1	7	3	7	6	6	7	-
Type	B	D	B	D	D	B	D	-

Figure 2: Sample sentence (morpheme-based). "Type" represents the type of dependency relation.

4 Algorithm

4.1 Dependency Representation

In our proposed algorithm, we use a morpheme-based dependency structure instead of a bunsetsu-based one. The morpheme-based representation is carefully designed to convey the same information on dependency structure of a sentence without the loss from the bunsetsu-based one. The rightmost morpheme of the bunsetsu t should modify the rightmost morpheme of the bunsetsu u when the bunsetsu t modifies the bunsetsu u. Every morpheme except the rightmost one in a bunsetsu should modify its following one. The sample sentence in Figure 1 is converted to the sentence with our proposed morpheme-based representation in Figure 2.

Take for instance, the head of the 0-th bunsetsu "Meg-ga" is the 4-th bunsetsu "age-ta." in Figure 1. This dependency relation is represented by that the head of the morpheme "ga" is "age-ta." in Figure 2.

The morpheme-based representation above cannot explicitly state the boundaries of bunsetsus. Thus we add the type to every dependency relation. A bunsetsu boundary is represented by the type associated with every dependency relation. The type "D" represents that this relation is a dependency of two bunsetsus, while the type "B" represents a sequence of morphemes inside of a given bunsetsu. In addition, the type "O", which represents that two morphemes do not have a dependency relation, is used in implementations of our algorithm with a trainable classifier. Following this encoding scheme of the type of dependency relations bunsetsu boundaries exist just after the morphemes that have the type "D". Inserting "|" after every morpheme with "D" of the sentence in Figure 2 results in Meg-ga | kare-ni | ano | pen-wo | age-ta. This is identical to the sentence with the bunsetsu-based representation in Figure 1.

Input: w_i: morphemes in a given sentence.
N: the number of morphemes.
Output: h_j: the head IDs of morphemes w_j.
t_j: the type of dependency relation. A possible value is either "B", "D", or "O".
Functions: Push(i, s): pushes i on the stack s.
Pop(s): pops a value off the stack s.
Dep(j, i, w, t): returns true when w_j should modify w_i. Otherwise returns false. Sets always t_j.

```
procedure Analyze(w, N, h, t)
var s: a stack for IDs of modifier morphemes
begin
  Push(−1, s);   { −1 for end-of-sentence }
  Push(0, s);
  for i ← 1 to N − 1 do begin
    j ← Pop(s);
    while (j ≠ −1
      and (Dep(j, i, w, t) or (i = N − 1)) ) do
    begin
      h_j ← i; j ← Pop(s)
    end
    Push(j, s); Push(i, s)
  end
end
```

Figure 3: Pseudo code for base phrase chunking and dependency parsing.

4.2 Pseudo Code for the Proposed Algorithm

The algorithm that we propose is based on (Sassano, 2004), which is considered to be a simple form of shift-reduce parsing. The pseudo code of our algorithm is presented in Figure 3. Important variables here are h_j and t_j where j is an index of morphemes. The variable h_j holds the head ID and the variable t_j has the type of dependency relation. For example, the head and the dependency relation type of "Meg" in Figure 2 are represented as $h_0 = 1$ and $t_0 =$ "B" respectively. The flow of the algorithm, which has the same structure as Sassano's (2004), is controlled with a stack that holds IDs for modifier morphemes. Decision of the relation between two morphemes is made in *Dep()*, which uses a machine learning-based classifier that supports multiclass prediction.

The presented algorithm runs in a left-to-right manner and its upper bound of the time complexity is $O(n)$. Due to space limitation, we do not discuss its complexity here. See (Sassano, 2004)

for further details.

5 Experiments and Discussion

5.1 Experimental Set-up

Corpus For evaluation, we used the Kyoto University Corpus Version 2 (Kurohashi and Nagao, 1998). The split for training/test/development is the same as in other papers, e.g., (Uchimoto et al., 1999).

Selection of a Classifier and its Setting We implemented a parser with the voted perceptron (VP) (Freund and Schapire, 1999). We used a polynomial kernel and set its degree to 3 because cubic kernels proved to be effective empirically for Japanese parsing (Kudo and Matsumoto, 2002). The number of epoch T of VP was selected using the development test set. For multiclass prediction, we used the pairwise method (Kreßel, 1999).

Features We have designed rather simple features based on the common feature set (Uchimoto et al., 1999; Kudo and Matsumoto, 2002; Sassano, 2004) for bunsetsu-based parsers. We use the following features for each morpheme:

1. major POS, minor POS, conjugation type, conjugation form, surface form (lexicalized form)
2. Content word or function word
3. Punctuation (periods and commas)
4. Open parentheses and close parentheses
5. Location (at the beginning or end of the sentence)

Gap features between two morphemes are also used since they have proven to be very useful and contribute to the accuracy (Uchimoto et al., 1999; Kudo and Matsumoto, 2002). They are represented as a binary feature and include distance (1, 2, 3, 4 – 10, or 11 $\leq$), particles, parentheses, and punctuation.

In our proposed algorithm basically two morphemes are examined to estimate their dependency relation. Context information about the current morphemes to be estimated would be very useful and we can incorporate such information into our model. We assume that we have the j-th morpheme and the i-th one in Figure 3. We also use the $j-n, ..., j-1, j+1, ..., j+n$ morphemes and the $i-n, ..., i-1, i+1, ..., i+n$ ones, where n is the size of the context window. We examined 0, 1, 2 and 3 for n.

Measure	Accuracy (%)
Dependency Acc.	93.96
Dep. Type Acc.	99.49
Both	93.92

Table 1: Performance on the test set. This result is achieved by the following parameters: The size of context window is 2 and epoch T is 4.

	Bunsetsu-based	Morpheme-based
Previous	88.48	95.09
Ours	NA	93.96

Table 2: Dependency accuracy. The system with the previous method employs the algorithm (Sassano, 2004) with the voted perceptron.

5.2 Results and Discussion

Accuracy Performances of our parser on the test set is shown in Table 1. The dependency accuracy is the percentage of the morphemes that have a correct head. The dependency type accuracy is the percentage of the morphemes that have a correct dependency type, i.e., "B" or "D". The bottom line of Table 1 shows the percentage of the morphemes that have both a correct head and a correct dependency type. In all these measures we excluded the last morpheme in a sentence, which does not have a head and its associated dependency type.

The accuracy of dependency type in Table 1 is interpreted to be accuracy of base phrase (bunsetsu) chunking. Very accurate chunking is achieved.

Next we examine the dependency accuracy. In order to recognize how accurate it is, we compared the performance of our parser with that of the parser that uses one of previous methods. We implemented a parser that employs the algorithm of (Sassano, 2004) with the commonly used features and runs with VP instead of SVM, which Sassano (2004) originally used. His parser, which cannot do bunsetsu chunking, accepts only a chunked sentence and then produces a bunsetsu-based dependency structure. Thus we cannot directly compare results with ours. To enable us to compare them we gave bunsetsu chunked sentences by our parser to the parser of (Sassano, 2004) instead of giving directly the correct chunked sentences

Window Size	Dep. Acc.	Dep. Type Acc.
0 ($T = 1$)	82.71	99.29
1 ($T = 2$)	93.57	99.49
2 ($T = 4$)	93.96	99.49
3 ($T = 3$)	93.79	99.42

Table 3: Performance change depending on the context window size

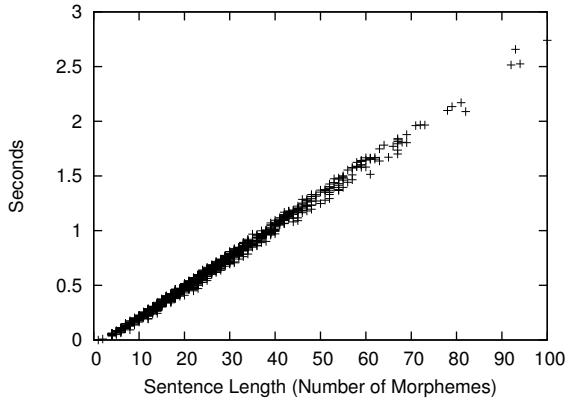


Figure 4: Running time on the test set. We used a PC (Intel Xeon 2.33 GHz with 8GB memory on FreeBSD 6.3).

in the Kyoto University Corpus. And then we received results from the parser of (Sassano, 2004), which are bunsetsu-based dependency structures, and converted them to morpheme-based structures that follow the scheme we propose in this paper. Finally we have got results that have the compatible format and show a comparison with them in Table 2.

Although the bunsetsu-based parser outperformed slightly our morpheme-based parser in this experiment, it is still notable that our method yields comparable performance with even a single scan of a sentence for dependency parsing in addition to bunsetsu chunking. According to the results in Table 2, we suppose that performance of our parser roughly corresponds to about 86–87% in terms of bunsetsu-based accuracy.

Context Window Size Performance change depending on the size of context window is shown in Table 3. Among them the best size is 2. In this case, we use ten morphemes to determine whether or not given two morphemes have a dependency relation. That is, to decide the relation of morphemes j and i $(j < i)$, we use morphemes $j-2, j-1, j, j+1, j+2$ and $i-2, i-1, i, i+1, i+2$.

Running Time and Asymptotic Time Complexity We have observed that the running time is proportional to the sentence length (Figure 4). The theoretical time complexity of the proposed algorithm is confirmed with this observation.

6 Conclusion and Future Work

We have described a novel algorithm that combines Japanese base phrase chunking and dependency parsing into a single scan process. The proposed algorithm runs in linear-time with a single scan of a sentence.

In future work we plan to combine morphological analysis or word segmentation into our proposed algorithm. We also expect that structure analysis of compound nouns can be incorporated by extending the dependency relation types. Furthermore, we believe it would be interesting to discuss linguistically and psycholinguistically the differences between Japanese and other European languages such as English. We would like to know what differences lead to easiness of analyzing a Japanese sentence.

References

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of CoNLL 2006*, pages 149–164.

Y. Freund and R. E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296.

U. Kreßel. 1999. Pairwise classification and support vector machines. In B. Schölkopf, C. J. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 255–268. MIT Press.

T. Kudo and Y. Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proc. of CoNLL-2002*, pages 63–69.

S. Kurohashi and M. Nagao. 1998. Building a Japanese parsed corpus while improving the parsing system. In *Proc. of LREC-1998*, pages 719–724.

S. Mori, M. Nishimura, N. Itoh, S. Ogino, and H. Watanabe. 2000. A stochastic parser based on a structural word prediction model. In *Proc. of COLING 2000*, pages 558–564.

S. Mori. 2002. A stochastic parser based on an SLM with arboreal context trees. In *Proc. of COLING 2002*.

M. Sassano. 2004. Linear-time dependency analysis for Japanese. In *Proc. of COLING 2004*, pages 8–14.

S. Sekine, K. Uchimoto, and H. Isahara. 2000. Backward beam search algorithm for dependency analysis of Japanese. In *Proc. of COLING-00*, pages 754–760.

K. Uchimoto, S. Sekine, and H. Isahara. 1999. Japanese dependency structure analysis based on maximum entropy models. In *Proc. of EACL-99*, pages 196–203.

Comparing the Accuracy of CCG and Penn Treebank Parsers

Stephen Clark
University of Cambridge
Computer Laboratory
15 JJ Thomson Avenue, Cambridge, UK
stephen.clark@cl.cam.ac.uk

James R. Curran
School of Information Technologies
University of Sydney
NSW 2006, Australia
james@it.usyd.edu.au

Abstract

We compare the CCG parser of Clark and Curran (2007) with a state-of-the-art Penn Treebank (PTB) parser. An accuracy comparison is performed by converting the CCG derivations into PTB trees. We show that the conversion is extremely difficult to perform, but are able to fairly compare the parsers on a representative subset of the PTB test section, obtaining results for the CCG parser that are statistically no different to those for the Berkeley parser.

1 Introduction

There are a number of approaches emerging in statistical parsing. The first approach, which began in the mid-90s and now has an extensive literature, is based on the Penn Treebank (PTB) parsing task: inferring skeletal phrase-structure trees for unseen sentences of the WSJ, and evaluating accuracy according to the Parseval metrics. Collins (1999) is a seminal example. The second approach is to apply statistical methods to parsers based on linguistic formalisms, such as HPSG, LFG, TAG, and CCG, with the grammar being defined manually or extracted from a formalism-specific treebank. Evaluation is typically performed by comparing against predicate-argument structures extracted from the treebank, or against a test set of manually annotated grammatical relations (GRs). Examples of this approach include Riezler et al. (2002), Miyao and Tsujii (2005), Briscoe and Carroll (2006), and Clark and Curran (2007).[1]

Despite the many examples from both approaches, there has been little comparison across the two groups, which we refer to as PTB parsing and formalism-based parsing, respectively. The PTB parser we use for comparison is the publicly available Berkeley parser (Petrov and Klein, 2007). The formalism-based parser we use is the CCG parser of Clark and Curran (2007), which is based on CCGbank (Hockenmaier and Steedman, 2007), a CCG version of the Penn Treebank. We compare this parser with a PTB parser because both are derived from the same original source, and both produce phrase-structure in some form or another; the interesting question is whether anything is gained by converting the PTB into CCG.[2]

The comparison focuses on accuracy and is performed by converting CCG derivations into PTB phrase-structure trees. A contribution of this paper is to demonstrate the difficulty of mapping from a grammatical resource based on the PTB back to the PTB, and we also comment on the (non-)suitability of the PTB as a general formalism-independent evaluation resource. A second contribution is to provide the first accuracy comparison of the CCG parser with a PTB parser, obtaining competitive scores for the CCG parser on a representative subset of the PTB test sections. It is important to note that the purpose of this evaluation is *comparison* with a PTB parser, rather than evaluation of the CCG parser *per se*. The CCG parser has been extensively evaluated elsewhere (Clark and Curran, 2007), and arguably GRs or predicate-argument structures provide a more suitable test set for the CCG parser than PTB phrase-structure trees.

2 The CCG to PTB Conversion

There has been much recent work in attempting to convert native parser output into alternative representations for evaluation purposes, e.g. (Clark and Curran, 2007; Matsuzaki and Tsujii, 2008). The conclusion is that such conversions are surprisingly difficult. Clark and Curran (2007)

[1] A third approach is dependency parsing, but we restrict the comparison in this paper to phrase-structure parsers.

[2] Since this short paper reports a small, focused research contribution, we refer readers to Clark and Curran (2007) and Petrov and Klein (2007) for details of the two parsers.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 53–56,
Suntec, Singapore, 4 August 2009.

shows that converting *gold-standard* CCG derivations into the GRs in DepBank resulted in an F-score of only 85%; hence the upper bound on the performance of the CCG parser, using this evaluation scheme, was only 85%. Given that the current best scores for the PTB parsing task are over 90%, any loss from the conversion process needs to be considered carefully if a fair comparison with PTB parsers is to be achieved.

CCGbank was derived from the PTB, and so it might be considered that converting back to the PTB would be a relatively easy task, by essentially reversing the mapping Hockenmaier and Steedman (2007) used to create CCGbank. However, there are a number of differences between the two treebanks which make the conversion back far from trivial. First, the corresponding derivations in the treebanks are not isomorphic: a CCG derivation is not simply a relabelling of the nodes in the PTB tree; there are many constructions, such as coordination and control structures, where the trees are a different shape, as well as having different labels. It is important to realise that Hockenmaier and Steedman (2007) invested a significant amount of time and effort in creating the mapping. Second, some of the labels in the PTB do not appear in CCGbank, for example the QP label, and these must be added back in; however, developing rules to insert these labels in the right places is a far from trivial task.

There were two approaches we considered for the conversion. One possibility is to associate PTB tree structures with CCG lexical categories, and combine the trees together in step with the category combinations in a CCG derivation — in much the same way that an LTAG has elementary trees in the lexicon which are combined using the substitution and adjunction rules of TAG. The second approach is to associate conversion rules with each local tree – i.e. a parent and one or two child nodes – which appears in the CCGbank data.[3] In this paper we took the second approach.

2.1 Conversion Schemas

There are three types of conversion schema: schemas which introduce nodes for lexical items; schemas which insert or elide PTB nodes for unary rules and type-raising; and schemas which can perform arbitrary manipulation of generated PTB subtrees for binary CCG rule instances. Examples of these schemas are shown in Table 1. The primary operations in the binary schema are inserting and attaching. Inserting a new node, for example using the schema `(S l r)`, creates a new `S` node dominating both the left and right children of a binary rule. The attaching schema can attach the left node under the right node (>); or the right node under the left node (<).

TYPE	RULE	SCHEMA
lexical	NP	`NP`
lexical	$NP[nb]/N$	–
lexical	$(S[dcl]\backslash NP)/NP$	`VP`
unary	$S[dcl] \rightarrow NP\backslash NP$	`(SBAR l)`
type-raising	$PP \rightarrow (S\backslash NP)\backslash((S\backslash NP)/PP)$	`l`
binary	$NP[nb]/N\ N \rightarrow NP[nb]$	`>`
binary	$NP\ S[dcl]\backslash NP \rightarrow S[dcl]$	`(S l r)`
binary	$NP/(S[dcl]\backslash NP)$ $S[dcl]\backslash NP \rightarrow NP$	`(SBAR l (S r))`

Table 1: Example conversion schemas

The lexical categories NP and $(S[dcl]\backslash NP)/NP$ (shown in Table 1) introduce the PTB nodes `NP` and `VP`, respectively, while other lexical categories such as $NP[nb]/N$ introduce no extra nodes. Some unary rules introduce nodes, such as `SBAR` for the reduced relative case, whilst others, such as the type-raised PP, do not. Finally, binary schemas may create no new nodes (e.g. when a determiner is attached to an existing NP), or one or more nodes (e.g. an extra `S` node is created when a verb phrase finds its subject).

A PTB tree is built from a CCG derivation by running over the derivation in a bottom-up fashion and applying these schemas to the local trees in the derivation.

2.2 Schema development

The schemas were developed by manual inspection using section 00 of CCGbank and the PTB as a development set, following the oracle methodology of Clark and Curran (2007), in which gold-standard derivations from CCGbank are converted to the new representation and compared with the gold standard for that representation. As well as giving an idea of the difficulty, and success, of the conversion, the resulting numbers provide an up-

[3] Another possible approach has been taken by Matsuzaki and Tsujii (2008), who convert HPSG analyses from a grammar automatically extracted from the PTB back into the PTB. They treat the problem as one of translation, learning a synchronous grammar to perform the mapping.

SECTION	P	R	F	COMP
00 (all)	93.37	95.15	94.25	39.68
00 (len $\leq$ 40)	94.11	95.65	94.88	42.11
23 (all)	93.68	95.13	94.40	39.93
23 (len $\leq$ 40)	93.75	95.23	94.48	42.15

Table 2: Oracle conversion evaluation

per bound on the performance of the CCG parser. The test set, section 23, was not inspected at any stage in the development of the schemas.

In total, we annotated 32 unary and 776 binary rule instances (of the possible 2853 instances) with conversion schemas, and 162 of the 425 lexical categories. We also implemented a small number of default catch-all cases for the general CCG combinatory rules and for the rules dealing with punctuation, which allowed most of the 2853 rule instances to be covered. Considerable time and effort was invested in the creation of these schemas.

The oracle conversion results from the gold standard CCGbank to the PTB for section 00 and 23 are shown in Table 2. The numbers are bracketing precision, recall, F-score and complete sentence matches, using the EVALB evaluation script. Note that these figures provide an upper bound on the performance of the CCG parser using EVALB, given the current conversion process.

The importance of this upper bound should not be underestimated, when the evaluation framework is such that incremental improvements of a few tenths of a percent are routinely presented as improving the state-of-the-art, as is the case with the Parseval metrics. The fact that the upper bound here is less than 95% shows that it is not possible to fairly evaluate the CCG parser on the complete test set. Even an upper bound of around 98%, which is achieved by Matsuzaki and Tsujii (2008), is not sufficient, since this guarantees a loss of at least 2%.[4]

3 Evaluation

The Berkeley parser (Petrov and Klein, 2007) provides performance close to the state-of-the-art for the PTB parsing task, with reported F-scores of around 90%. Since the oracle score for CCGbank is less than 95%, it would not be a fair comparison to use the complete test set. However, there are a number of sentences which are correct, or almost correct, according to EVALB after the conversion, and we are able to use those for a fair comparison.

Table 3 gives the EVALB results for the CCG parser on various subsets of section 00 of the PTB. The first row shows the results on only those sentences which the conversion process can convert sucessfully (as measured by converting gold-standard CCGbank derivations and comparing with PTB trees; although, to be clear, the scores are for the CCG parser on those sentences). As can be seen from the scores, these sentences form a slightly easier subset than the full section 00, but this is a subset which can be used for a fair comparison against the Berkeley parser, since the conversion process is not lossy for this subset.

The second row shows the scores on those sentences for which the conversion process was somewhat lossy, but when the gold-standard CCGbank derivations are converted, the oracle F-measure is greater than 95%. The third row is similar, but for sentences for which the oracle F-score is geater than 92%. The final row is for the whole of section 00. The UB column gives the upper bound on the accuracy of the CCG parser. Results are calculated using both gold standard and automatically assigned POS tags; # is the number of sentences in the sample, and the % column gives the sample size as a percentage of the whole section.

We compare the CCG parser to the Berkeley parser using the accurate mode of the Berkeley parser, together with the model supplied with the publicly available version. Table 3 gives the results for Section 23, comparing the CCG and Berkeley parsers. The projected columns give the projected scores for the CCG parser, if it performed at the same accuracy level for those sentences which could not be converted successfully. The purpose of this column is to obtain an approximation of the CCG parser score for a perfect conversion process.[5] The results in bold are those which we consider to be a fair comparison against the Berkeley parser. The difference in scores is not statistically significant at p=0.05 (using Dan Bikel's stratified shuffling test).

One possible objection to this comparison is that the subset for which we have a fair compar-

[4] The higher upper bound achieved by Matsuzaki and Tsujii (2008) could be due to the fact that their extracted HPSG grammars are closer to the PTB than CCGbank, or due to their conversion method. We leave the application of their method to the CCG parser for future work.

[5] This is likely to be an upper bound on the performance of the CCG parser, since the larger test sets contain sentences which were harder to convert, and hence are likely to be more difficult to parse.

SAMPLE	#	%	UB	actual F		projected F	
				gold	auto	gold	auto
00 (F=100)	759	39.7	100.00	94.19	93.41	–	–
00 (F≥95)	1164	60.8	98.49	91.08	89.93	92.46	91.29
00 (F≥92)	1430	74.6	97.41	89.73	88.47	92.05	90.76
00 (all)	1913	100.0	94.25	87.00	85.60	92.00	90.52

Table 3: Results on the development set (CCG parser only)

SAMPLE	#	%	UB	Berkeley F		actual F		projected F	
				gold	auto	gold	auto	gold	auto
23 (F=100)	961	39.9	100.0	**93.38**	**93.37**	**93.83**	**92.86**	–	–
23 (F≥95)	1401	58.2	98.61	91.66	91.63	90.82	89.84	92.08	91.09
23 (F≥92)	1733	72.0	97.44	91.01	90.88	89.53	88.54	91.82	90.81
23 (all)	2407	100.0	94.40	89.67	89.47	86.36	85.50	91.20	90.29

Table 4: Results on the test set (CCG parser and Berkeley)

ison is likely to be an easy subset consisting of shorter sentences, and so the most that can be said is that the CCG parser performs as well as the Berkeley parser on short sentences. In fact, the subset for which we perform a perfect conversion contains sentences with an average length of 18.1 words, compared to 21.4 for sentences with 40 words or less (a standard test set for reporting Parseval figures). Hence we do consider the comparison to be highly informative.

4 Conclusion

One question that is often asked of the CCG parsing work is "Why not convert back into the PTB representation and perform a Parseval evaluation?" By showing how difficult the conversion is, we believe that we have finally answered this question, as well as demonstrating comparable performance with the Berkeley parser. In addition, we have thrown further doubt on the possible use of the PTB for cross-framework parser evaluation, as recently suggested by Matsuzaki and Tsujii (2008). Even the smallest loss due to mapping across representations is significant when a few tenths of a percentage point matter. Whether PTB parsers could be competitive on alternative parser evaluations, such as those using GR schemes, for which the CCG parser performs very well, is an open question.

Acknowledgements

James Curran was funded under Australian Research Council Discovery grant DP0665973. Stephen Clark was funded under EPSRC grant EP/E035698/1.

References

Ted Briscoe and John Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *Proceedings of the Poster Session of COLING/ACL-06*, Sydney, Austrailia.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Takuya Matsuzaki and Jun'ichi Tsujii. 2008. Comparative parser performance analysis across grammar frameworks through automatic tree conversion using synchronous grammars. In *Proceedings of COLING-08*, pages 545–552, Manchester, UK.

Yusuke Miyao and Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd meeting of the ACL*, pages 83–90, University of Michigan, Ann Arbor.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of the HLT/NAACL conference*, Rochester, NY.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell III, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proceedings of the 40th Meeting of the ACL*, pages 271–278, Philadelphia, PA.

A Framework for Entailed Relation Recognition

Dan Roth Mark Sammons V.G.Vinod Vydiswaran
University of Illinois at Urbana-Champaign
{danr|mssammon|vgvinodv}@illinois.edu

Abstract

We define the problem of recognizing entailed relations – given an open set of relations, find **all** occurrences of the relations of interest in a given document set – and pose it as a challenge to scalable information extraction and retrieval. Existing approaches to relation recognition do not address well problems with an open set of relations and a need for high recall: supervised methods are not easily scaled, while unsupervised and semi-supervised methods address a limited aspect of the problem, as they are restricted to frequent, explicit, highly localized patterns. We argue that textual entailment (*TE*) is necessary to solve such problems, propose a scalable TE architecture, and provide preliminary results on an Entailed Relation Recognition task.

1 Introduction

In many information foraging tasks, there is a need to find all text snippets relevant to a target concept. Patent search services spend significant resources looking for prior art relevant to a specified patent claim. Before subpoenaed documents are used in a court case or intelligence data is declassified, all sensitive sections need to be redacted. While there may be a specific domain for a given application, the set of target concepts is broad and may change over time. For these knowledge-intensive tasks, we contend that feasible automated solutions require techniques which approximate an appropriate level of natural language understanding.

Such problems can be formulated as a relation recognition task, where the information need is expressed as tuples of arguments and relations. This structure provides additional information which can be exploited to precisely fulfill the information need. Our work introduces the *Entailed Relation Recognition* paradigm, which leverages a textual entailment system to try to extract *all* relevant passages for a given structured query without requiring relation-specific training data. This contrasts with Open Information Extraction (Banko and Etzioni, 2008) and On-Demand Information Extraction (Sekine, 2006), which aim to extract large databases of open-ended facts, and with supervised relation extraction, which requires additional supervised data to learn new relations.

Specifically, the contributions of this paper are: 1. Introduction of the *entailed relation recognition* framework; 2. Description of an architecture and a system which uses structured queries and an existing entailment engine to perform relation extraction; 3. Empirical assessment of the system on a corpus of entailed relations.

2 Entailed Relation Recognition (ERR)

In the task of Entailed Relation Recognition, a corpus and an information need are specified. The corpus comprises all text spans (e.g. paragraphs) contained in a set of documents. The information need is expressed as a set of tuples encoding relations and entities of interest, where entities can be of arbitrary type. The objective is to retrieve ***all*** relevant text spans that a human would recognize as containing a relation of interest. For example:

Information Need: An organization acquires weapons.

Text 1**:** ...the recent theft of 500 assault rifles by FARC...

Text 2**:** ...the report on FARC activities made three main observations. First, their allies supplied them with the 3" mortars used in recent operations. Second, ...

Text 3**:** Amnesty International objected to the use of artillery to drive FARC militants from heavily populated areas.

An automated system should identify Texts 1 and 2 as containing the relation of interest, and Text 3 as irrelevant. The system must therefore detect relation instances that cross sentence boundaries ("them" maps to "FARC", Text 2), and that require inference ("theft" implies "acquire", Text 1). It must also discern when sentence structure precludes a match ("Amnesty International... use... artillery" does not imply "Amnesty International

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 57–60,
Suntec, Singapore, 4 August 2009.

acquires artillery", Text 3).

The problems posed by instances like Text 2 are beyond the scope of traditional unsupervised and semi-supervised relation-extraction approaches such as those used by Open IE and On-Demand IE, which are constrained by their dependency on limited, sentence-level structure and high-frequency, highly local patterns, in which relations are explicitly expressed as verbs and nouns. Supervised methods such as (Culotta and Sorensen, 2004) and (Roth and Yih, 2004) provide only a partial solution, as there are many possible relations and entities of interest for a given domain, and such approaches require new annotated data each time a new relation or entity type is needed. Information Retrieval approaches are optimized for document-level performance, and enhancements like pseudo-feedback (Rocchio, 1971) are less applicable to the localized text spans needed in the tasks of interest; as such, it is unlikely that they will reliably retrieve all correct instances, and not return superficially similar but incorrect instances (such as Text 3) with high rank.

Attempts have been made to apply Textual Entailment in larger scale applications. For the task of Question Answering, (Harabagiu and Hickl, 2006) applied a TE component to rerank candidate answers returned by a retrieval step. However, QA systems rely on redundancy in the same way Open IE does: a large document set has so many instances of a given relation that at least some will be sufficiently explicit and simple that standard IR approaches will retrieve them. A single correct instance suffices to complete the QA task, but does not meet the needs of the task outlined here.

Recognizing relation instances requiring inference steps, in the absence of labeled training data, requires a level of text understanding. A suitable proxy for this would be a successful Textual Entailment Recognition (*TE*) system. (Dagan et al., 2006) define the task of Recognizing Textual Entailment (*RTE*) as: *...a directional relation between two text fragments, termed T – the entailing text, and H – the entailed text. T entails H if, typically, a human reading T would infer that H is most likely true.* For relation recognition, the relation triple (e.g. "Organization acquires weapon") is the hypothesis, and a candidate text span that might contain the relation is the text. The definition of RTE clearly accommodates the range of phenomena described for the examples above. However, the more successful TE systems (e.g. (Hickl and Bensley, 2007)) are typically resource intensive, and cannot scale to large retrieval tasks if a brute force approach is used.

We define the task of Entailed Relation Recognition thus: *Given a text collection D, and an information need specified in a set of [argument, relation, argument] triples S: for each triple $s \in S$, identify all texts $d \in D$ such that d entails s.*
The information need triples, or queries, encode relations between arbitrary entities (specifically, these are *not* constrained to be Named Entities).

This problem is distinct from recent work in Textual Entailment as we constrain the structure of the Hypothesis to be very simple, and we require that the task be of a significantly larger scale than the RTE tasks to date (which are typically of the order of 800 Text-Hypothesis pairs).

3 Scalable ERR Algorithm

Our scalable ERR approach, *SERR*, consists of two stages: expanded lexical retrieval, and entailment recognition. The SERR algorithm is presented in Fig. 1. The goal is to scale Textual Entailment up to a task involving large corpora, where hypotheses (queries) may be entailed by multiple texts. The task is kept tractable by decomposing TE capabilities into two steps.

The first step, Expanded Lexical Retrieval (*ELR*), uses shallow semantic resources and similarity measures, thereby incorporating some of the semantic processing used in typical TE systems. This is required to retrieve, with high recall, semantically similar content that may not be lexically similar to query terms, to ensure return of a set of texts that are highly likely to contain the concept of interest.

The second step applies a textual entailment system to this text set and the query in order to label the texts as 'relevant' or 'irrelevant', and requires deeper semantic resources in order to discern texts containing the concept of interest from those that do not. This step emphasizes higher precision, as it filters irrelevant texts.

3.1 Implementation of SERR

In the ELR stage, we use a structured query that allows more precise search and differential query expansion for each query element. Semantic units in the texts (e.g. Named Entities, phrasal verbs) are indexed separately from words; each index is

SERR Algorithm

SETUP:

Input: Text set D
Output: Indices $\{I\}$ over D
for all texts $d \in D$
    Annotate d with local semantic content
Build Search Indices $\{I\}$ over D

APPLICATION:

Input: Information need S

EXPANDED LEXICAL RETRIEVAL (ELR)(s):

$R \leftarrow \emptyset$
Expand s with semantically similar words
Build search query q_s from s
$R \leftarrow k$ top-ranked texts for q_s using indices $\{I\}$
return R

SERR:

Answer set $A \leftarrow \emptyset$
for all queries $s \in S$
    $R \leftarrow$ ELR(s)
    Answer set $A_s \leftarrow \emptyset$
    for all results $r \in R$
        Annotate s, r with NLP resources
        if r entails s
            $A_s \leftarrow A_s \cup r$
    $A \leftarrow A \cup \{A_s\}$
return A

Figure 1. SERR algorithm

a hierarchical similarity structure based on a type-specific metric (e.g. WordNet-based for phrasal verbs). Query structure is also used to selectively expand query terms using similarity measures related to types of semantic units, including distributional similarity (Lin and Pantel, 2001), and measures based on WordNet (Fellbaum, 1998).

We assess three different Textual Entailment components: **LexPlus**, a lexical-level system that achieves relatively good performance on the RTE challenges, and two variants of Predicate-based Textual Entailment, **PTE-strict** and **PTE-relaxed**, which use a predicate-argument representation. The former is constrained to select a single predicate-argument structure from each result, which is compared to the query component-by-component using similarity measures similar to the LexPlus system. PTE-relaxed drops the single-predicate constraint, and can be thought of as a 'bag-of-constituents' model. In both, features are extracted based on the predicate-argument components' match scores and their connecting structure, and the rank assigned by ELR. These features are used by a classifier that labels each result as 'relevant' or 'irrelevant'. Training examples are selected from the top 7 results returned by ELR for queries corresponding to entailment pair hypotheses from the RTE development corpora; test examples are similarly selected from results for queries from the RTE test corpora (see section 3.2).

3.2 Entailed Relation Recognition Corpus

To assess performance on the ERR task, we derive a corpus from the publicly available RTE data. The corpus consists of a set S of information needs in the form of [argument, relation, argument] triples, and a set D of text spans (short paragraphs), half of which entail one or more $s \in S$ while the other half are unrelated to S. D comprises all $1,950$ Texts from the *IE* and *IR* subtasks of the RTE Challenge 1–3 datasets. The shorter hypotheses in these examples allow us to automatically induce their structured query form from their shallow semantic structure. S was automatically generated from the positive entailment pairs in D, by annotating their hypotheses with a publicly available SRL tagger (Punyakanok et al., 2008) and inferring the relation and two main arguments to form the equivalent queries.

Since some Hypotheses and Texts appear multiple times in the RTE corpora, we automatically extract mappings from positive Hypotheses to one or more Texts by comparing hypotheses and texts from different examples. This provides the labeling needed for evaluation. In the resulting corpus, a wide range of relations are sparsely represented; they exemplify many linguistic and semantic characteristics required to infer the presence of non-explicit relations.

4 Results and Discussion

Top #	Basic	ELR	Rel.Impr.	Err.Redu.
1	48.1%	55.2%	+14.8%	13.7%
2	68.1%	72.8%	+6.9%	14.7%
3	75.2%	78.5%	+4.4%	17.7%

Table 1. Change in relevant results retrieved in top 3 positions for basic and expanded lexical retrieval

System	Acc.	Prec.	Rec.	F_1
Baseline	18.1	18.1	100.0	30.7
LexPlus	81.6	44.9	62.5	55.5
PTE-relax.	71.9	37.7	72.0	49.0
	(0.1)	(5.5)	(6.2)	(4.1)
PTE-strict	**83.6**	55.4	61.5	**57.9**
	(1.3)	(3.4)	(7.9)	(2.1)

Table 2. Comparison of performance of SERR with different TE algorithms. Numbers in parentheses are standard deviations.

Table 1 compares the results of SERR with and

# System	RTE 1	RTE 2	RTE 3	Avg. Acc.
LexPlus	49.0	65.2 [3]	76.5 [2]	**66.3**
PTE-relaxed	54.5 (1.0)	68.7 (1.5) [3]	82.3 (2.0) [1]	**71.2 (1.2)**
PTE-strict	64.8 (2.3) [1]	71.2 (2.6) [3]	76.0 (3.2) [2]	**71.8 (2.6)**

Table 3. Performance (accuracy) of SERR system variants on RTE challenge examples; numbers in parentheses are standard deviations, while numbers in brackets indicate where systems would have ranked in the RTE evaluations.

	Comparisons
Standard TE	3,802,500
SERR	13,650

Table 4. Entailment comparisons needed for standard TE vs. SERR

without the ELR's semantic enhancements. For each rank k, the entries represent the proportion of queries for which the correct answer was returned in the top k positions. The semantic enhancements improve the number of matched results at each of the top 3 positions.

Table 2 compares variants of the SERR implementation. The baseline labels every result returned by ELR as 'relevant', giving high recall but low precision. PTE-relaxed performs better than baseline, but poorly compared to PTE-strict and LexPlus. Our analysis shows that LexPlus has a relatively high threshold, and correctly labels as negative some examples mislabeled by PTE-relaxed, which may match two of the three constituents in a hypothesis and label that result as positive. PTE-strict will correctly identify some such examples as it will force some match edges to be ignored, and will correctly identify some negative examples due to structural constraints even when LexPlus finds matches for all query terms. PTE-strict strikes the best balance between precision and recall on positive examples.

Table 3 shows the accuracy of SERR's classification of the examples from each RTE challenge; results not returned in the top 7 ranks by ELR are labeled 'irrelevant'. PTE-strict and PTE-relaxed perform comparably overall, though PTE-strict has more uniform results over the different challenges. Both outperform the LexPlus system overall, and perform well compared to the best results published for the RTE challenges.

The significant computational gain of SERR is shown in Table 4, exhibiting the much greater number of comparisons required by a brute force TE approach compared to SERR: SERR performs well compared to published results for RTE challenges 1-3, but makes only 0.36% of the TE comparisons needed by standard approaches on our ERR task.

5 Conclusion

We have proposed an approach to solving the Entailed Relation Recognition task, based on Textual Entailment, and implemented a solution that shows that a Textual Entailment Recognition system can be scaled to a much larger IE problem than that represented by the RTE challenges. Our preliminary results demonstrate the utility of the proposed architecture, which allows strong performance in the RTE task and efficient application to a large corpus (table 4).

Acknowledgments

We thank Quang Do, Yuancheng Tu, and Kevin Small. This work is funded by a grant from Boeing and by MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

References

[Banko and Etzioni2008] M. Banko and O. Etzioni. 2008. The Tradeoffs Between Open and Traditional Relation Extraction. In *ACL-HLT*, pages 28–36.

[Culotta and Sorensen2004] A. Culotta and J. Sorensen. 2004. Dependency Tree Kernels for Relation Extraction. In *ACL*, pages 423–429.

[Dagan et al.2006] I. Dagan, O. Glickman, and B. Magnini, editors. 2006. *The PASCAL Recognising Textual Entailment Challenge.*, volume 3944. Springer-Verlag, Berlin.

[Fellbaum1998] C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

[Harabagiu and Hickl2006] S. Harabagiu and A. Hickl. 2006. Methods for Using Textual Entailment in Open-Domain Question Answering. In *ACL*, pages 905–912.

[Hickl and Bensley2007] A. Hickl and J. Bensley. 2007. A Discourse Commitment-Based Framework for Recognizing Textual Entailment. In *ACL*, pages 171–176.

[Lin and Pantel2001] D. Lin and P. Pantel. 2001. Induction of semantic classes from natural language text. In *SIGKDD*, pages 317–322.

[Punyakanok et al.2008] V. Punyakanok, D. Roth, and W. Yih. 2008. The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. *CL*, 34(2).

[Rocchio1971] J. Rocchio, 1971. *Relevance feedback in Information Retrieval*, pages 313–323. Prentice Hall.

[Roth and Yih2004] D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *CoNLL*, pages 1–8.

[Sekine2006] S. Sekine. 2006. On-Demand Information Extraction. In *COLING/ACL*, pages 731–738.

A Combination of Active Learning and Semi-supervised Learning Starting with Positive and Unlabeled Examples for Word Sense Disambiguation: An Empirical Study on Japanese Web Search Query

Makoto Imamura and Yasuhiro Takayama
Information Technology R&D Center,
Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura, Kanagawa, Japan
{Imamura.Makoto@bx,Takayama.Yasuhiro@ea}.MitsubishiElectric.co.jp

Nobuhiro Kaji, Masashi Toyoda and Masaru Kitsuregawa
Institute of Industrial Science,
The University of Tokyo
4-6-1 Komaba, Meguro-ku Tokyo, Japan
{kaji,toyoda,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract

This paper proposes to solve the bottleneck of finding training data for word sense disambiguation (WSD) in the domain of web queries, where a complete set of ambiguous word senses are unknown. In this paper, we present a combination of active learning and semi-supervised learning method to treat the case when positive examples, which have an expected word sense in web search result, are only given. The novelty of our approach is to use "pseudo negative examples" with reliable confidence score estimated by a classifier trained with positive and unlabeled examples. We show experimentally that our proposed method achieves close enough WSD accuracy to the method with the manually prepared negative examples in several Japanese Web search data.

1 Introduction

In Web mining for sentiment or reputation analysis, it is important for reliable analysis to extract large amount of texts about certain products, shops, or persons with high accuracy. When retrieving texts from Web archive, we often suffer from word sense ambiguity and WSD system is indispensable. For instance, when we try to analyze reputation of "Loft", a name of variety store chain in Japan, we found that simple text search retrieved many unrelated texts which contain "Loft" with different senses such as an attic room, an angle of golf club face, a movie title, a name of a club with live music and so on. The words in Web search queries are often proper nouns. Then it is not trivial to discriminate these senses especially for the language like Japanese whose proper nouns are not capitalized.

To train WSD systems we need a large amount of positive and negative examples. In the real Web mining application, how to acquire training data for a various target of analysis has become a major hurdle to use supervised WSD.

Fortunately, it is not so difficult to create positive examples. We can retrieve positive examples from Web archive with high precision (but low recall) by manually augmenting queries with hypernyms or semantically related words (e.g., "Loft AND shop" or "Loft AND stationary").

On the other hand, it is often costly to create negative examples. In principle, we can create negative examples in the same way as we did to create positive ones. The problem is, however, that we are not sure of most of the senses of a target word. Because target words are often proper nouns, their word senses are rarely listed in hand-crafted lexicon. In addition, since the Web is huge and contains heterogeneous domains, we often find a large number of unexpected senses. For example, all the authors did not know the music club meaning of Loft. As the result, we often had to spend much time to find such unexpected meaning of target words.

This situation motivated us to study active learning for WSD starting with only positive examples. The previous techniques (Chan and Ng, 2007; Chen et al. 2006) require balanced positive and negative examples to estimate the score. In our problem setting, however, we have no negative examples at the initial stage. To tackle this problem, we propose a method of active learning for WSD with pseudo negative examples, which are selected from unlabeled data by a classifier trained with positive and unlabeled examples. McCallum and Nigam (1998) combined active learning and semi-supervised learning technique

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 61–64,
Suntec, Singapore, 4 August 2009.

by using EM with unlabeled data integrated into active learning, but it did not treat our problem setting where only positive examples are given.

The construction of this paper is as follows; Section 2 describes a proposed learning algorithm. Section 3 shows the experimental results.

2 Learning Starting with Positive and Unlabeled Examples for WSD

We treat WSD problem as binary classification where desired texts are positive examples and other texts are negative examples. This setting is practical, because ambiguous senses other than the expected sense are difficult to know and are no concern in most Web mining applications.

2.1 Classifier

For our experiment, we use naive Bayes classifiers as learning algorithm. In performing WSD, the sense "s" is assigned to an example characterized with the probability of linguistic features $f_1,...,f_n$ so as to maximize:

$$p(s)\prod_{j=1}^{n} p(f_j \mid s) \qquad (1)$$

The sense s is positive when it is the target meaning in Web mining application, otherwise s is negative. We use the following typical linguistic features for Japanese sentence analysis, (a) Word feature within sentences, (b) Preceding word feature within bunsetsu (Japanese base phrase), (c) Backward word feature within bunsetsu, (d) Modifier bunsetsu feature and (e) Modifiee bunsetsu feature.

Using naive Bayes classifier, we can estimate the confidence score c(d, s) that the sense of a data instance "d", whose features are f_1, f_2, ..., f_n, is predicted sense "s".

$$c(d,s) = \log p(s) + \sum_{j=1}^{n} \log p(f_j \mid s) \qquad (2)$$

2.2 Proposed Algorithm

At the beginning of our algorithm, the system is provided with positive examples and unlabeled examples. The positive examples are collected by full text queries with hypernyms or semantically related words.

First we select positive dataset P from initial dataset by manually augmenting full text query.

At each iteration of active learning, we select pseudo negative dataset N_p (Figure 1 line 15). In selecting pseudo negative dataset, we predict word sense of each unlabeled example using the naive Bayes classifier with all the unlabeled examples as negative examples (Figure 2). In detail, if the prediction score (equation(3)) is more than τ, which means the example is very likely to be negative, it is considered as the pseudo negative example (Figure 2 line 10-12).

$$c(d, psdNeg) = c(d, neg) - c(d, pos) \qquad (3)$$

```
01  # Definition
02  Γ (P, N): WSD system trained on P as Positive
03          examples, N as Negative examples.
04  Γ EM(P, N, U): WSD system trained on P as
05      Positive examples, N as Negative examples,
06      U as Unlabeled examples by using EM
07      (Nigam et. all 2000)
08  # Input
09  T ← Initial unlabeled dataset which contain
10      ambiguous words
11  # Initialization
12  P ← positive training dataset by full text search on T
13  N ← ϕ (initial negative training dataset)
14  repeat
15   # selecting pseudo negative examples Np
16     by the score of Γ (P, T-P)   (see figure 2)
17   # building a classifier with Np
18    Γ new ← Γ EM (P, N+Np, T-N-P)
19   # sampling data by using the score of Γ new
20   cmin ← ∞
21   foreach d ∈ (T – P – N )
22     classify d by WSD system Γ new
23     s(d) ← word sense prediction for d using Γ new
24     c(d, s(d)) ← the confidence of prediction of d
25     if c(d, s(d)) < cmin then
26        cmin ← c(d),  d min ← d
27   end
28  end
29   provide correct sense s for d min by human
30   if s is positive then add d min to P
31                  else add d min to N
32  until Training dataset reaches desirable size
33  Γ new is the output classifier
```

Figure 1: A combination of active learning and semi-supervised learning starting with positive and unlabeled examples

Next we use Nigam's semi-supervised learning method using EM and a naive Bayes classifier (Nigam et. all, 2000) with pseudo negative dataset N_p as negative training dataset to build the refined classifier Γ_{EM} (Figure 1 line 17).

In building training dataset by active learning, we use uncertainty sampling like (Chan and Ng, 2007) (Figure 1 line 30-31). This step selects the most uncertain example that is predicted with the lowest confidence in the refined classifier Γ_{EM}. Then, the correct sense for the most uncertain

example is provided by human and added to the positive dataset P or the negative dataset N according to the sense of d.

The above steps are repeated until dataset reaches the predefined desirable size.

```
01  foreach d ∈ ( T – P – N )
02    classify d by WSD system Γ (P, T-P)
03    c(d, pos) ← the confidence score that d is
04       predicted as positive defined in equation (2)
05    c(d, neg) ← the confidence score that d is
06       predicted as negative defined in equation (2)
07    c(d, psdNeg) = c(d, neg) - c(d, pos)
08                (the confidence score that d is
09                 predicted as pseudo negative)
10     PN ← d ∈ ( T – P – N ) | s(d) = neg ∧
11                                c(d, psdNeg) ≧ τ }
12              (PN is pseudo negative dataset )
13  end
```

Figure 2: Selection of pseudo negative examples

3 Experimental Results

3.1 Data and Condition of Experiments

We select several example data sets from Japanese blog data crawled from Web. Table 1 shows the ambiguous words and each ambiguous senses.

Word	Positive sense	Other ambiguous senses
Wega	product name (TV)	Las Vegas, football team name, nickname, star, horse race, Baccarat glass, atelier, wine, game, music
Loft	store name	attic room, angle of golf club face, club with live music, movie
Honda	personal name (football player)	Personal names (actress, artists, other football players, etc.) hardware store, car company name
Tsubaki	product name (shampoo)	flower name, kimono, horse race, camellia ingredient, shop name

Table 1: Selected examples for evaluation

Table 2 shows the ambiguous words, the number of its senses, the number of its data instances, the number of feature, and the percentage of positive sense instances for each data set.

Assigning the correct labels of data instances is done by one person and 48.5% of all the labels are checked by another person. The percentage of agreement between 2 persons for the assigned labels is 99.0%. The average time of assigning labels is 35 minutes per 100 instances.

Selected instances for evaluation are randomly divided 10% test set and 90% training set. Table 3 shows the each full text search query and the number of initial positive examples and the percentage of it in the training data set.

word	No. of senses	No. of instances	No. of features	Percentage of positive sense
Wega	11	5,372	164,617	31.1%
Loft	5	1,582	38,491	39.4%
Honda	25	2,100	65,687	21.2%
Tsubaki	6	2,022	47,629	40.2%

Table 2: Selected examples for evaluation

word	Full text query for initial positive examples	No. of positive examples (percentage in trainig set)
Wega	Wega AND TV	316 (6.5%)
Loft	Loft AND (Grocery OR-Stationery)	64 (4.5%)
Honda	Honda AND Keisuke	86 (4.6%)
Tsubaki	Tsubaki AND Shiseido	380 (20.9%)

Table 3: Initial positive examples

The threshold value τ in figure 2 is set to empirically optimized value 50. Dependency on threshold value τ will be discussed in 3.3.

3.2 Comparison Results

Figure 3 shows the average WSD accuracy of the following 6 approaches.

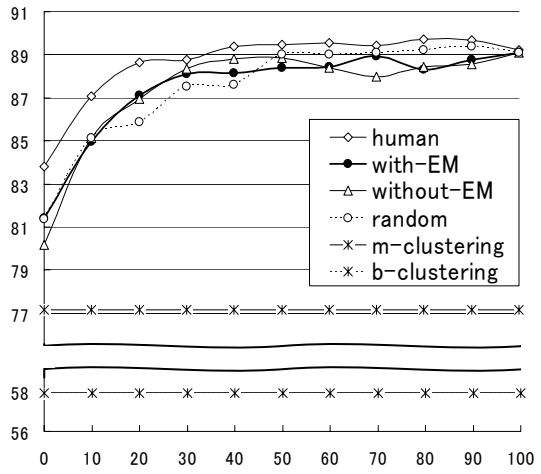


Figure 3: Average active learning process

B-clustering is a standard unsupervised WSD, a clustering using naive Bayes classifier learned with two cluster numbers via EM algorithm. The given number of the clusters are two, negative and positive datasets.

M-clustering is a variant of b-clustering where the given number of clusters are each number of ambiguous word senses in table 2.

Human labeling, abbreviated as *human*, is an active learning approach starting with human labeled negative examples. The number of hu-

man labeled negative examples in initial training data is the same as that of positive examples in figure 3. Human labeling is considered to be the upper accuracy in the variants of selecting pseudo negative examples.

Random sampling with EM, abbreviated as *with-EM*, is the variant approach where d_{min} in line 26 of figure 1 is randomly selected without using confidence score.

Uncertainty sampling without EM (Takayama et al. 2009), abbreviated as *without-EM*, is a variant approach where Γ_{EM} (P, N+N_p, T-N-P) in line 18 of figure 1 is replaced by Γ (P, N+N_p).

Uncertainty Sampling with EM, abbreviated as *uncertain*, is a proposed method described in figure 1.

The accuracy of the proposed approach *with-EM* is gradually increasing according to the percentage of added hand labeled examples.

The initial accuracy of *with-EM*, which means the accuracy with no hand labeled negative examples, is the best score 81.4% except for that of *human*. The initial WSD accuracy of *with-EM* is 23.4 and 4.2 percentage points higher than those of b-clustering (58.0%) and m-clustering (77.2%), respectively. This result shows that the proposed selecting method of pseudo negative examples is effective.

The initial WSD accuracy of *with-EM* is 1.3 percentage points higher than that of *without-EM* (80.1%). This result suggests semi-supervised learning using unlabeled examples is effective.

The accuracies of *with-EM, random* and *without-EM* are gradually increasing according to the percentage of added hand labeled examples and catch up that of *human* and converge at 30 percentage added points. This result suggests that our proposed approach can reduce the labor cost of assigning correct labels.

The curve *with-EM* are slightly upper than the curve *random* at the initial stage of active learning. At 20 percentage added point, the accuracy with-EM is 87.0 %, 1.1 percentage points higher than that of random (85.9%). This result suggests that the effectiveness of proposed uncertainty sampling method is not remarkable depending on the word distribution of target data.

There is really not much difference between the curve *with-EM* and *without-EM*. As a classifies to use the score for sampling examples in adaptation iterations, it is indifferent whether *with-EM* or *without-EM*.

Larger evaluation is the future issue to confirm if the above results could be generalized beyond the above four examples used as proper nouns.

3.3 Dependency on Threshold Value τ

Figure 4 shows the average WSD accuracies of *with-EM* at 0, 25, 50 and 75 as the values of τ. The each curve represents our proposed algorithm with threshold value τ in the parenthesis. The accuracy in the case of τ = 75 is higher than that of τ = 50 over 20 percentage data added point. This result suggests that as the number of hand labeled negative examples increasing, τ should be gradually decreasing, that is, the number of pseudo negative examples should be decreasing. Because, if sufficient number of hand labeled negative examples exist, a classifier does not need pseudo negative examples. The control of τ depending on the number of hand labeled examples during active learning iterations is a future issue.

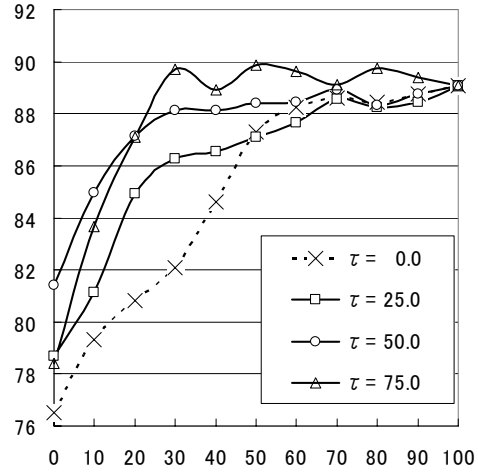


Figure 4: Dependency of threshold value τ

References

Chan, Y. S. and Ng, H. T. 2007. Domain Adaptation with Active Learning for Word Sense Disambiguation. *Proc. of ACL 2007*, 49-56.

Chen, J., Schein, A., Ungar, L., and Palmer, M. 2006. An Empirical Study of the Behavior of Active Learning for Word Sense Disambiguation, *Proc. of the main conference on Human Language Technology Conference of the North American Chapter of ACL*, pp. 120-127.

McCallum, A. and Nigam, K. 1998. Employing EM and Pool-Based Active Learning for Text Classification. *Proceedings of the Fifteenth international Conference on Machine Learning*, 350-358.

Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. 2000. Text Classification from Labeled and Unlabeled Documents using EM, *Machine Learning*, 39, 103-134.

Takayama, Y., Imamura, M., Kaji N., Toyoda, M. and Kitsuregawa, M. 2009. Active Learning with Pseudo Negative Examples for Word Sense Disambiguation in Web Mining (in Japanese), Journal of IPSJ (in printing).

Detecting Compositionality in Multi-Word Expressions

Ioannis Korkontzelos
Department of Computer Science
The University of York
Heslington, York, YO10 5NG, UK
johnkork@cs.york.ac.uk

Suresh Manandhar
Department of Computer Science
The University of York
Heslington, York, YO10 5NG, UK
suresh@cs.york.ac.uk

Abstract

Identifying whether a multi-word expression (*MWE*) is compositional or not is important for numerous *NLP* applications. Sense induction can partition the context of *MWEs* into semantic uses and therefore aid in deciding compositionality. We propose an unsupervised system to explore this hypothesis on *compound nominals*, *proper names* and *adjective-noun constructions*, and evaluate the contribution of sense induction. The evaluation set is derived from *WordNet* in a semi-supervised way. Graph connectivity measures are employed for unsupervised parameter tuning.

1 Introduction and related work

Multi-word expressions (*MWEs*) are sequences of words that tend to cooccur more frequently than chance and are either idiosyncratic or decomposable into multiple simple words (Baldwin, 2006). Deciding idiomaticity of *MWEs* is highly important for *machine translation*, *information retrieval*, *question answering*, *lexical acquisition*, *parsing* and *language generation*.

Compositionality refers to the degree to which the meaning of a *MWE* can be predicted by combining the meanings of its components. Unlike *syntactic compositionality* (e.g. *by and large*), *semantic compositionality* is continuous (Baldwin, 2006).

In this paper, we propose a novel unsupervised approach that compares the major senses of a *MWE* and its semantic head using distributional similarity measures to test the compositionality of the *MWE*. These senses are induced by a graph based sense induction system, whose parameters are estimated in an unsupervised manner exploiting a number of graph connectivity measures (Korkontzelos et al., 2009). Our method partitions the context space and only uses the major senses, filtering out minor senses. In our approach the only language dependent components are a *PoS* tagger and a parser.

There are several studies relevant to detecting compositionality of noun-noun *MWEs* (Baldwin et al., 2003) verb-particle constructions (Bannard et al., 2003; McCarthy et al., 2003) and verb-noun pairs (Katz and Giesbrecht, 2006). Datasets with human compositionality judgements are available for these *MWE* categories (Cook et al., 2008). Here, we focus on *compound nominals*, *proper names* and *adjective-noun constructions*.

Our contributions are three-fold: firstly, we experimentally show that sense induction can assist in identifying compositional *MWEs*. Secondly, we show that unsupervised parameter tuning (Korkontzelos et al., 2009) results in accuracy that is comparable to the best manually selected combination of parameters. Thirdly, we propose a semi-supervised approach for extracting non-compositional *MWEs* from *WordNet*, to decrease annotation cost.

2 Proposed approach

Let us consider the non-compositional *MWE* "red carpet". It mainly refers to a strip of red carpeting laid down for dignitaries to walk on. However, it is possible to encounter instances of "red carpet" referring to any carpet of red colour. Our method first applies sense induction to identify the major semantic uses (senses) of a *MWE* ("red carpet") and its semantic head ("carpet"). Then, it compares these uses to decide *MWE* compositionality. The more diverse these uses are, the more possibly the *MWE* is non-compositional. Our algorithm consists of 4 steps:

A. Corpora collection and preprocessing. Our approach receives as input a *MWE* (e.g. "red carpet"). The dependency output of *Stanford Parser* (Klein and Manning, 2003) is used to locate the

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 65–68,
Suntec, Singapore, 4 August 2009.

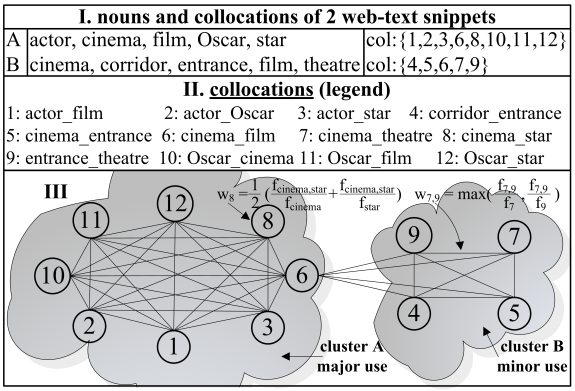


Figure 1: "red carpet", sense induction example

MWE semantic head. Two different corpora are collected (for the *MWE* and its semantic head). Each consists of webtext snippets of length 15 to 200 tokens in which the *MWE*/semantic head appears. Given a *MWE*, a set of queries is created: All synonyms of the *MWE* extracted from WordNet are collected[1]. The *MWE* is paired with each synonym to create a set of queries. For each query, snippets are collected by parsing the web-pages returned by *Yahoo!*. The union of all snippets produces the *MWE* corpus. The corpus for a semantic head is created equivalently.

To keep the computational time reasonable, only the longest $3,000$ snippets are kept from each corpus. Both corpora are *PoS* tagged (*GENIA* tagger). In common with Agirre et al. (2006), only nouns are kept and lemmatized, since they are more discriminative than other *PoS*.

B. Sense Induction methods can be broadly divided into vector-space models and graph based models. Sense induction methods are evaluated under the *SemEval-2007* framework (Agirre and Soroa, 2007). We employ the collocational graph-based sense induction of Klapaftis and Manandhar (2008) in this work (henceforth referred to as KM). The method consists of 3 stages:

Corpus preprocessing aims to capture nouns that are contextually related to the target *MWE*/head. *Log-likelihood ratio* (G^2) (Dunning, 1993) with respect to a large reference corpus, *Web 1T 5-gram Corpus* (Brants and Franz, 2006), is used to capture the contextually relevant nouns. P_1 is the G^2 threshold below which nouns are removed from corpora.

Graph creation. A collocation is defined as a pair of nouns cooccuring within a snippet. Each noun within a snippet is combined with every other, generating $\binom{n}{2}$ collocations. Each collocation is represented as a weighted vertex. P_2 thresholds collocation frequencies and P_3 collocation weights. Weighted edges are drawn based on cooccurrence of the corresponding vertices in one or more snippets (e.g. w_8 and $w_{7,9}$, fig. 1). In contrast to KM, frequencies for weighting vertices and edges are obtained from *Yahoo!* web-page counts to deal with data sparsity.

Graph clustering uses *Chinese Whispers*[2] (Biemann, 2006) to cluster the graph. Each cluster now represents a sense of the target word.

KM produces larger number of clusters (uses) than expected. To reduce it we exploit the *one sense per collocation* property (Yarowsky, 1995). Given a cluster l_i, we compute the set S_i of snippets that contain at least one collocation of l_i. Any clusters l_a and l_b are merged if $S_a \subseteq S_b$.

C. Comparing the induced senses. We used two techniques to measure the distributional similarity of major uses of the *MWE* and its semantic head, both based on *Jaccard coefficient (J)*. "Major use" denotes the cluster of collocations which tags the most snippets. Lee (1999) shows that *J* performs better than other symmetric similarity measures such as *cosine*, *Jensen-Shannon divergence*, etc. The first is $J_c = J(A,B) = \frac{|A \cap B|}{|A \cup B|}$, where A, B are sets of collocations. The second, J_{sn}, is based on the snippets that are tagged by the induced uses. Let K_i be the set of snippets in which at least one collocation of the use i occurs. $J_{sn} = J(K_j, K_k)$, where j, k are the major uses of the *MWE* and its semantic head, respectively.

D. Determining compositionality. Given the major uses of a *MWE* and its semantic head, the *MWE* is considered as compositional, when the corresponding distributional similarity measure (J_c or J_{sn}) value is above a parameter threshold, *sim*. Otherwise, it is considered as non-compositional.

3 Test set of *MWEs*

To the best of our knowledge there are no noun compound datasets accompanied with compositionality judgements available. Thus, we developed an algorithm to aid human annotation. For each of the $52,217$ *MWEs* of *WordNet 3.0* (Miller, 1995) we collected:

[1] Thus, for "red carpet", corpora will be collected for "red carpet" and "carpet". The synonyms of "red carpet" are "rug", "carpet" and "carpeting"

[2] *Chinese Whispers* is not guaranteed to converge, thus 200 was adopted as the maximum number of iterations.

Non-compositional *MWEs*
agony aunt, ***black maria***, ***dead end***, *dutch oven*, ***fish finger***, *fool's paradise*, ***goat's rue***, ***green light***, ***high jump***, joint chiefs, ***lip service***, ***living rock***, ***monkey puzzle***, ***motor pool***, ***prince Albert***, ***stocking stuffer***, ***sweet bay***, **teddy boy**, ***think tank***
Compositional *MWEs*
box white oak, **cartridge brass**, ***common iguana***, closed chain, eastern pipistrel, field mushroom, **hard candy**, ***king snake***, ***labor camp***, lemon tree, ***life form***, parenthesis-free notation, ***parking brake***, ***petit juror***, **relational adjective**, *taxonomic category*, ***telephone service***, tea table, ***upland cotton***

Table 1: Test set with compositionality annotation. *MWEs* whose compositionality was successfully detected by: (a) *1c1word* baseline are in **bold** font, (b) manual parameter selection are underlined and (c) *average cluster coefficient* are in *italics*.

1. all synonyms of the *MWE*
2. all hypernyms of the *MWE*
3. sister-synsets of the *MWE*, within distance[3] 3
4. synsets that are in holonymy or meronymy relation to the *MWE*, within distance 3

If the semantic head of the *MWE* is also in the above collection then the *MWE* is likely to be compositional, otherwise it is likely that the *MWE* is non-compositional.

$6,287$ *MWEs* were judged as potentially non-compositional. We randomly chose 19 and checked them manually. Those that were compositional were replaced by other randomly chosen ones. The process was repeated until we ended up with 19 non-compositional examples. Similarly, 19 negative examples that were judged as compositional were collected (Table 1).

4 Evaluation setting and results

The sense induction component of our algorithm depends upon 3 parameters: P_1 is the G^2 threshold below which noun are removed from corpora. P_2 thresholds collocation frequencies and P_3 collocation weights. We chose $P_1 \in \{5, 10, 15\}$, $P_2 \in \{10^2, 10^3, 10^4, 10^5\}$ and $P_3 \in \{0.2, 0.3, 0.4\}$. For reference, P_1 values of 3.84, 6.63, 10.83 and 15.13 correspond to G^2 values for confidence levels of 95%, 99%, 99.9% and 99.99%, respectively.

To assess the performance of the proposed algorithm we compute *accuracy*, the percentage of *MWEs* whose compositionality was correctly determined against the gold standard.

[3]Locating sister synsets at distance D implies ascending D steps and then descending D steps.

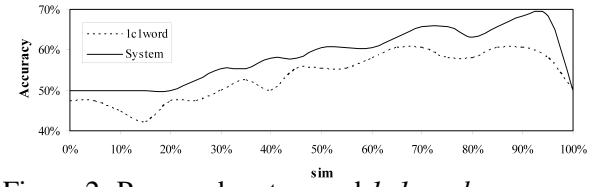


Figure 2: Proposed system and *1c1word* accuracy.

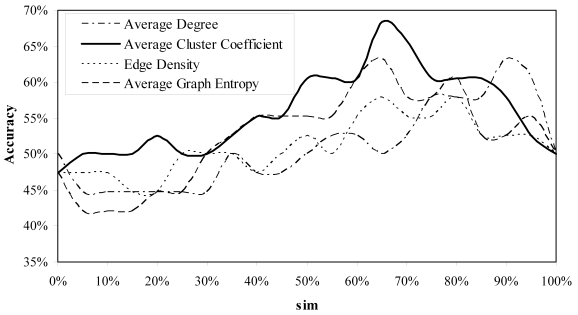


Figure 3: Unweighted graph con/vity measures.

We compared the system's performance against a baseline, *1c1word*, that assigns the whole graph to a single cluster and no graph clustering is performed. *1c1word* corresponds to a relevant *SemEval-2007* baseline (Agirre and Soroa, 2007) and helps in showing whether sense induction can assist determining compositionality.

Our method was evaluated for each $\langle P_1, P_2, P_3 \rangle$ combination and similarity measures J_c and J_{sn}, separately. We used our development set to determine if there are parameter values that verify our hypothesis. Given a *sim* value (see section 2, last paragraph), we chose the best performing parameter combination manually.

The best results for manual parameter selection were obtained for $sim = 95\%$ giving an accuracy of 68.42% for detecting non-compositional MWEs. In all experiments, J_{sn} outperforms J_c. With manually selected parameters, our system's accuracy is higher than *1c1word* for all sim values (5% points) (fig. 2, table 1). The initial hypothesis holds; *sense induction* improves *MWE* compositionality detection.

5 Unsupervised parameter tuning

We followed Korkontzelos et al. (2009) to select the "best" parameters $\langle P_1, P_2, P_3 \rangle$ for the collocational graph of each *MWE* or head word. We applied 8 graph connectivity measures (weighted and unweighted versions of *average degree*, *cluster coefficient*, *graph entropy* and *edge density*) separately on each of the clusters (resulting from the application of the chinese whispers algorithm).

Each graph connectivity measure assigns a score to each cluster. We averaged the scores over

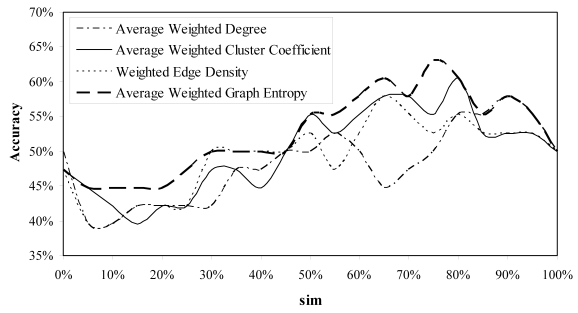


Figure 4: Weighted graph connectivity measures.

the clusters from the same graph. For each connectivity measure, we chose the parameter combination $\langle P_1, P_2, P_3 \rangle$ that gave the highest score.

While manual parameter tuning chooses a single globally best set of parameters (see section 4), the graph connectivity measures generate different values of $\langle P_1, P_2, P_3 \rangle$ for each graph.

5.1 Evaluation results

The best performing distributional similarity measure is J_{sn}. Unweighted versions of graph connectivity measures perform better than weighted ones. Figures 3 and 4 present a comparison between the unweighted and weighted versions of all *graph connectivity measures*, respectively, for all *sim* values. *Average cluster coefficient* performs better or equally well to the other *graph connectivity measures* for all sim values (except for $sim \in [90\%, 100\%]$). The accuracy of *average cluster coefficient* is equal (68.42%) to that of manual parameter selection (section 4, table 1). The second best performing unweighted *graph connectivity measures* is *average graph entropy*. For weighted *graph connectivity measures*, *average graph entropy* performs best, followed by *average weighted clustering coefficient*.

6 Conclusion and Future Work

We hypothesized that sense induction can assist in identifying compositional *MWEs*. We introduced an unsupervised system to experimentally explore the hypothesis, and showed that it holds. We proposed a semi-supervised way to extract non-compositional *MWEs* from *WordNet*. We showed that graph connectivity measures can be successfully employed to perform unsupervised parameter tuning of our system. It would be interesting to explore ways to substitute querying *Yahoo!* so as to make the system quicker. Experimentation with more sophisticated graph connectivity measures could possibly improve accuracy.

References

E. Agirre and A. Soroa. 2007. Semeval-2007, task 02: Evaluating WSI and discrimination systems. In *proceedings of SemEval-2007*. ACL.

E. Agirre, D. Martínez, O. de Lacalle, and A. Soroa. 2006. Two graph-based algorithms for state-of-the-art WSD. In *proceedings of EMNLP-2006*. ACL.

T. Baldwin, C. Bannard, T. Tanaka, and D. Widdows. 2003. An empirical model of MWE decomposability. In *proceedings of the MWE workshop*. ACL.

T. Baldwin. 2006. Compositionality and MWEs: Six of one, half a dozen of the other? In *proceedings of the MWE workshop*. ACL.

C. Bannard, T. Baldwin, and A. Lascarides. 2003. A statistical approach to the semantics of verb-particles. In *proceedings of the MWE workshop*. ACL.

C. Biemann. 2006. Chinese whispers - an efficient graph clustering algorithm and its application to NLP problems. In *proceedings of TextGraphs*. ACL.

T. Brants and A. Franz. 2006. Web 1t 5-gram corpus, version 1. Technical report, Google Research.

P. Cook, A. Fazly, and S. Stevenson. 2008. The VNC-Tokens Dataset. In *proceedings of the MWE workshop*. ACL.

T. Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

G. Katz and E. Giesbrecht. 2006. Automatic identification of non-compositional MWEs using latent semantic analysis. In *proceedings of the MWE workshop*. ACL.

I. P. Klapaftis and S. Manandhar. 2008. WSI using graphs of collocations. In *proceedings of ECAI-2008*.

D. Klein and C. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *proceedings of NIPS 15*. MIT Press.

I. Korkontzelos, I. Klapaftis, and S. Manandhar. 2009. Graph connectivity measures for unsupervised parameter tuning of graph-based sense induction systems. In *proceedings of the UMSLLS Workshop, NAACL HLT 2009*.

L. Lee. 1999. Measures of distributional similarity. In *proceedings of ACL*.

D. McCarthy, B. Keller, and J. Carroll. 2003. Detecting a continuum of compositionality in phrasal verbs. In *proceedings of the MWE workshop*. ACL.

G. A. Miller. 1995. WordNet: a lexical database for English. *ACM*, 38(11):39–41.

D. Yarowsky. 1995. Unsupervised WSD rivaling supervised methods. In *proceedings of ACL*.

Directional Distributional Similarity for Lexical Expansion

Lili Kotlerman, Ido Dagan, Idan Szpektor
Department of Computer Science
Bar-Ilan University
Ramat Gan, Israel
lili.dav@gmail.com
{dagan,szpekti}@cs.biu.ac.il

Maayan Zhitomirsky-Geffet
Department of Information Science
Bar-Ilan University
Ramat Gan, Israel
maayan.geffet@gmail.com

Abstract

Distributional word similarity is most commonly perceived as a symmetric relation. Yet, one of its major applications is lexical expansion, which is generally asymmetric. This paper investigates the nature of directional (asymmetric) similarity measures, which aim to quantify distributional feature inclusion. We identify desired properties of such measures, specify a particular one based on averaged precision, and demonstrate the empirical benefit of directional measures for expansion.

1 Introduction

Much work on automatic identification of semantically similar terms exploits Distributional Similarity, assuming that such terms appear in similar contexts. This has been now an active research area for a couple of decades (Hindle, 1990; Lin, 1998; Weeds and Weir, 2003).

This paper is motivated by one of the prominent applications of distributional similarity, namely identifying lexical expansions. Lexical expansion looks for terms whose meaning implies that of a given target term, such as a query. It is widely employed to overcome lexical variability in applications like Information Retrieval (IR), Information Extraction (IE) and Question Answering (QA). Often, distributional similarity measures are used to identify expanding terms (e.g. (Xu and Croft, 1996; Mandala et al., 1999)). Here we denote the relation between an expanding term u and an expanded term v as '$u \rightarrow v$'.

While distributional similarity is most prominently modeled by symmetric measures, lexical expansion is in general a directional relation. In IR, for instance, a user looking for "baby food" will be satisfied with documents about "baby pap" or "baby juice" ('pap $\rightarrow$ food', 'juice $\rightarrow$ food'); but when looking for "frozen juice" she will not be satisfied by "frozen food". More generally, directional relations are abundant in NLP settings, making symmetric similarity measures less suitable for their identification.

Despite the need for directional similarity measures, their investigation counts, to the best of our knowledge, only few works (Weeds and Weir, 2003; Geffet and Dagan, 2005; Bhagat et al., 2007; Szpektor and Dagan, 2008; Michelbacher et al., 2007) and is utterly lacking. From an expansion perspective, the common expectation is that the context features characterizing an expanding word should be largely included in those of the expanded word.

This paper investigates the nature of directional similarity measures. We identify their desired properties, design a novel measure based on these properties, and demonstrate its empirical advantage in expansion settings over state-of-the-art measures[1]. In broader prospect, we suggest that asymmetric measures might be more suitable than symmetric ones for many other settings as well.

2 Background

The distributional word similarity scheme follows two steps. First, a feature vector is constructed for each word by collecting context words as features. Each feature is assigned a weight indicating its "relevance" (or association) to the given word. Then, word vectors are compared by some vector similarity measure.

[1] Our directional term-similarity resource will be available at http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 69–72,
Suntec, Singapore, 4 August 2009.

To date, most distributional similarity research concentrated on symmetric measures, such as the widely cited and competitive (as shown in (Weeds and Weir, 2003)) *LIN* measure (Lin, 1998):

$$LIN(u,v) = \frac{\sum_{f \in FV_u \cap FV_v}[w_u(f) + w_v(f)]}{\sum_{f \in FV_u} w_u(f) + \sum_{f \in FV_v} w_v(f)}$$

where FV_x is the feature vector of a word x and $w_x(f)$ is the weight of the feature f in that word's vector, set to their pointwise mutual information.

Few works investigated a directional similarity approach. Weeds and Weir (2003) and Weeds et al. (2004) proposed a *precision* measure, denoted here *WeedsPrec*, for identifying the hyponymy relation and other generalization/specification cases. It quantifies the weighted coverage (or *inclusion*) of the candidate hyponym's features (u) by the hypernym's (v) features:

$$WeedsPrec(u \rightarrow v) = \frac{\sum_{f \in FV_u \cap FV_v} w_u(f)}{\sum_{f \in FV_u} w_u(f)}$$

The assumption behind *WeedsPrec* is that if one word is indeed a generalization of the other then the features of the more specific word are likely to be included in those of the more general one (but not necessarily vice versa).

Extending this rationale to the textual entailment setting, Geffet and Dagan (2005) expected that if the meaning of a word u entails that of v then *all* its prominent context features (under a certain notion of "prominence") would be included in the feature vector of v as well. Their experiments indeed revealed a strong empirical correlation between such complete inclusion of prominent features and lexical entailment, based on web data. Yet, such complete inclusion cannot be feasibly assessed using an off-line corpus, due to the huge amount of required data.

Recently, (Szpektor and Dagan, 2008) tried identifying the entailment relation between lexical-syntactic templates using *WeedsPrec*, but observed that it tends to promote unreliable relations involving infrequent templates. To remedy this, they proposed to balance the directional *WeedsPrec* measure by multiplying it with the symmetric *LIN* measure, denoted here *balPrec*:

$$balPrec(u \rightarrow v) = \sqrt{LIN(u,v) \cdot WeedsPrec(u \rightarrow v)}$$

Effectively, this measure penalizes infrequent templates having short feature vectors, as those usually yield low symmetric similarity with the longer vectors of more common templates.

3 A Statistical Inclusion Measure

Our research goal was to develop a directional similarity measure suitable for learning asymmetric relations, focusing empirically on lexical expansion. Thus, we aimed to quantify most effectively the above notion of *feature inclusion*.

For a candidate pair 'u → v', we will refer to the set of u's features, which are those tested for inclusion, as *tested features*. Amongst these features, those found in v's feature vector are termed *included features*.

In preliminary data analysis of pairs of feature vectors, which correspond to a known set of valid and invalid expansions, we identified the following desired properties for a distributional inclusion measure. Such measure should reflect:

1. the proportion of included features amongst the tested ones (the core inclusion idea).

2. the relevance of included features to the expanding word.

3. the relevance of included features to the expanded word.

4. that inclusion detection is less reliable if the number of features of either expanding or expanded word is small.

3.1 Average Precision as the Basis for an Inclusion Measure

As our starting point we adapted the *Average Precision* (*AP*) metric, commonly used to score ranked lists such as query search results. This measure combines precision, relevance ranking and overall recall (Voorhees and Harman, 1999):

$$AP = \frac{\sum_{r=1}^{N}[P(r) \cdot rel(r)]}{\textit{total number of relevant documents}}$$

where r is the rank of a retrieved document amongst the N retrieved, $rel(r)$ is an indicator function for the relevance of that document, and $P(r)$ is precision at the given cut-off rank r.

In our case the feature vector of the expanded word is analogous to the set of all relevant documents while tested features correspond to retrieved documents. Included features thus correspond to relevant retrieved documents, yielding the follow-

ing analogous measure in our terminology:

$$AP(u \rightarrow v) = \frac{\sum_{r=1}^{|FV_u|}[P(r) \cdot rel(f_r)]}{|FV_v|}$$

$$rel(f) = \begin{cases} 1, & \text{if } f \in FV_v \\ 0, & \text{if } f \notin FV_v \end{cases}$$

$$P(r) = \frac{|\textit{included features in ranks } 1 \textit{ to } r|}{r}$$

where f_r is the feature at rank r in FV_u.

This analogy yields a feature inclusion measure that partly addresses the above desired properties. Its score increases with a larger number of included features (correlating with the 1st property), while giving higher weight to highly ranked features of the expanding word (2nd property).

To better meet the desired properties we introduce two modifications to the above measure. First, we use the number of tested features $|FV_u|$ for normalization instead of $|FV_v|$. This captures better the notion of feature inclusion (1st property), which targets the proportion of included features relative to the *tested* ones.

Second, in the classical *AP* formula all relevant documents are considered relevant to the same extent. However, features of the expanded word differ in their relevance within its vector (3rd property). We thus reformulate $rel(f)$ to give higher relevance to highly ranked features in $|FV_v|$:

$$rel'(f) = \begin{cases} 1 - \frac{rank(f, FV_v)}{|FV_v|+1} & \text{, if } f \in FV_v \\ 0 & \text{, if } f \notin FV_v \end{cases}$$

where $rank(f, FV_v)$ is the rank of f in FV_v.

Incorporating these two modifications yields the *APinc* measure:

$$\textit{APinc}(u \rightarrow v) = \frac{\sum_{r=1}^{|FV_u|}[P(r) \cdot rel'(f_r)]}{|FV_u|}$$

Finally, we adopt the balancing approach in (Szpektor and Dagan, 2008), which, as explained in Section 2, penalizes similarity for infrequent words having fewer features (4th property) (in our version, we truncated *LIN* similarity lists after top 1000 words). This yields our proposed directional measure *balAPinc*:

$$\textit{balAPinc}(u \rightarrow v) = \sqrt{\textit{LIN}(u, v) \cdot \textit{APinc}(u \rightarrow v)}$$

4 Evaluation and Results

4.1 Evaluation Setting

We tested our similarity measure by evaluating its utility for lexical expansion, compared with baselines of the *LIN*, *WeedsPrec* and *balPrec* measures (Section 2) and a balanced version of *AP* (Section 3), denoted *balAP*. Feature vectors were created by parsing the Reuters RCV1 corpus and taking the words related to each term through a dependency relation as its features (coupled with the relation name and direction, as in (Lin, 1998)). We considered for expansion only terms that occur at least 10 times in the corpus, and as features only terms that occur at least twice.

As a typical lexical expansion task we used the ACE 2005 events dataset[2]. This standard IE dataset specifies 33 event types, such as *Attack*, *Divorce*, and *Law Suit*, with all event mentions annotated in the corpus. For our lexical expansion evaluation we considered the first IE subtask of finding sentences that mention the event.

For each event we specified a set of representative words (*seeds*), by selecting typical terms for the event (4 on average) from its ACE definition. Next, for each similarity measure, the terms found similar to any of the event's seeds ('$u \rightarrow$ *seed*') were taken as expansion terms. Finally, to measure the sole contribution of expansion, we removed from the corpus all sentences that contain a seed word and then extracted all sentences that contain expansion terms as mentioning the event. Each of these sentences was scored by the sum of similarity scores of its expansion terms.

To evaluate expansion quality we compared the ranked list of sentences for each event to the gold-standard annotation of event mentions, using the standard Average Precision (AP) evaluation measure. We report Mean Average Precision (MAP) for all events whose AP value is at least 0.1 for at least one of the tested measures[3].

4.1.1 Results

Table 1 presents the results for the different tested measures over the ACE experiment. It shows that the symmetric *LIN* measure performs significantly worse than the directional measures, assessing that a directional approach is more suitable for the expansion task. In addition, balanced measures consistently perform better than unbalanced ones.

According to the results, *balAPinc* is the best-performing measure. Its improvement over all other measures is statistically significant according to the two-sided Wilcoxon signed-rank test

[2] http://projects.ldc.upenn.edu/ace/, training part.

[3] The remaining events seemed useless for our comparative evaluation, since suitable expansion lists could not be found for them by any of the distributional methods.

LIN	*WeedsPrec*	*balPrec*	*AP*	*balAP*	*balAPinc*
0.068	0.044	0.237	0.089	0.202	**0.312**

Table 1: MAP scores of the tested measures on the ACE experiment.

seed	*LIN*	*balAPinc*
death	murder, killing, incident, arrest, violence	suicide, killing, fatality, murder, mortality
marry	divorce, murder, love, dress, abduct	divorce, remarry, father, kiss, care for
arrest	detain, sentence, charge, jail, convict	detain, extradite, round up, apprehend, imprison
birth	abortion, pregnancy, resumption, seizure, passage	wedding day, dilation, birthdate, circumcision, triplet
injure	wound, kill, shoot, detain, burn	wound, maim, beat up, stab, gun down

Table 2: Top 5 expansion terms learned by *LIN* and *balAPinc* for a sample of ACE seed words.

(Wilcoxon, 1945) at the 0.01 level. Table 2 presents a sample of the top expansion terms learned for some ACE seeds with either *LIN* or *balAPinc*, demonstrating the more accurate expansions generated by *balAPinc*. These results support the design of our measure, based on the desired properties that emerged from preliminary data analysis for lexical expansion.

Finally, we note that in related experiments we observed statistically significant advantages of the *balAPinc* measure for an unsupervised text categorization task (on the 10 most frequent categories in the Reuters-21578 collection). In this setting, category names were taken as seeds and expanded by distributional similarity, further measuring cosine similarity with categorized documents similarly to IR query expansion. These experiments fall beyond the scope of this paper and will be included in a later and broader description of our work.

5 Conclusions and Future work

This paper advocates the use of directional similarity measures for lexical expansion, and potentially for other tasks, based on distributional inclusion of feature vectors. We first identified desired properties for an inclusion measure and accordingly designed a novel directional measure based on averaged precision. This measure yielded the best performance in our evaluations. More generally, the evaluations supported the advantage of multiple directional measures over the typical symmetric *LIN* measure.

Error analysis showed that many false sentence extractions were caused by ambiguous expanding and expanded words. In future work we plan to apply disambiguation techniques to address this problem. We also plan to evaluate the performance of directional measures in additional tasks, and compare it with additional symmetric measures.

Acknowledgements

This work was partially supported by the NEGEV project (www.negev-initiative.org), the PASCAL-2 Network of Excellence of the European Community FP7-ICT-2007-1-216886 and by the Israel Science Foundation grant 1112/08.

References

R. Bhagat, P. Pantel, and E. Hovy. 2007. LEDIR: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of EMNLP-CoNLL*.

M. Geffet and I. Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of ACL*.

D. Hindle. 1990. Noun classification from predicate-argument structures. In *Proceedings of ACL*.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*.

R. Mandala, T. Tokunaga, and H. Tanaka. 1999. Combining multiple evidence from different types of thesaurus for query expansion. In *Proceedings of SIGIR*.

L. Michelbacher, S. Evert, and H. Schutze. 2007. Asymmetric association measures. In *Proceedings of RANLP*.

I. Szpektor and I. Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of COLING*.

E. M. Voorhees and D. K. Harman, editors. 1999. *The Seventh Text REtrieval Conference (TREC-7)*, volume 7. NIST.

J. Weeds and D. Weir. 2003. A general framework for distributional similarity. In *Proceedings of EMNLP*.

J. Weeds, D. Weir, and D. McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of COLING*.

F. Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83.

J. Xu and W. B. Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of SIGIR*.

Generalizing over Lexical Features: Selectional Preferences for Semantic Role Classification

Beñat Zapirain, Eneko Agirre
Ixa Taldea
University of the Basque Country
Donostia, Basque Country
{benat.zapirain,e.agirre}@ehu.es

Lluís Màrquez
TALP Research Center
Technical University of Catalonia
Barcelona, Catalonia
lluism@lsi.upc.edu

Abstract

This paper explores methods to alleviate the effect of lexical sparseness in the classification of verbal arguments. We show how automatically generated selectional preferences are able to generalize and perform better than lexical features in a large dataset for semantic role classification. The best results are obtained with a novel second-order distributional similarity measure, and the positive effect is specially relevant for out-of-domain data. Our findings suggest that selectional preferences have potential for improving a full system for Semantic Role Labeling.

1 Introduction

Semantic Role Labeling (SRL) systems usually approach the problem as a sequence of two subtasks: argument *identification* and *classification*. While the former is mostly a syntactic task, the latter requires semantic knowledge to be taken into account. Current systems capture semantics through lexicalized features on the predicate and the head word of the argument to be classified. Since lexical features tend to be sparse (especially when the training corpus is small) SRL systems are prone to overfit the training data and generalize poorly to new corpora.

This work explores the usefulness of selectional preferences to alleviate the lexical dependence of SRL systems. Selectional preferences introduce semantic generalizations on the type of arguments preferred by the predicates. Therefore, they are expected to improve generalization on infrequent and unknown words, and increase the discriminative power of the argument classifiers.

For instance, consider these two sentences:

JFK was assassinated (in Dallas)$_{Location}$
JFK was assassinated (in November)$_{Temporal}$

Both share syntactic and argument structure, so the lexical features (i.e., the words 'Dallas' and 'November') represent the most important knowledge to discriminate between the two different adjunct roles. The problem is that, in new text, one may encounter similar expressions with new words like Texas or Autumn.

We propose a concrete classification problem as our main evaluation setting for the acquired selectional preferences: *given a verb occurrence and a nominal head word of a constituent dependant on that verb, assign the most plausible role to the head word according to the selectional preference model*. This problem is directly connected to argument classification in SRL, but we have isolated the evaluation from the complete SRL task. This first step allows us to analyze the potential of selectional preferences as a source of semantic knowledge for discriminating among different role labels. Ongoing work is devoted to the integration of selectional preference–derived features in a complete SRL system.

2 Related Work

Automatic acquisition of selectional preferences is a relatively old topic, and will mention the most relevant references. Resnik (1993) proposed to model selectional preferences using semantic classes from WordNet in order to tackle ambiguity issues in syntax (noun-compounds, coordination, PP-attachment).

Brockman and Lapata (2003) compared several *class-based* models (including Resnik's selectional preferences) on a syntactic plausibility judgement task for German. The models return weights for (verb, syntactic_function, noun) triples, and the correlation with human plausibility judgement is used for evaluation. Resnik's selectional preference scored best among class-based methods, but it performed equal to a simple, purely lexical, conditional probability model.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 73–76,
Suntec, Singapore, 4 August 2009.

Distributional similarity has also been used to tackle syntactic ambiguity. Pantel and Lin (2000) obtained very good results using the distributional similarity measure defined by Lin (1998).

The application of selectional preferences to semantic roles (as opposed to syntactic functions) is more recent. Gildea and Jurafsky (2002) is the only one applying selectional preferences in a real SRL task. They used distributional clustering and WordNet-based techniques on a SRL task on FrameNet roles. They report a very small improvement of the overall performance when using distributional clustering techniques. In this paper we present complementary experiments, with a different role set and annotated corpus (PropBank), a wider range of selectional preference models, and the analysis of out-of-domain results.

Other papers applying semantic preferences in the context of semantic roles, rely on the evaluation on pseudo tasks or human plausibility judgments. In (Erk, 2007) a distributional similarity–based model for selectional preferences is introduced, reminiscent of that of Pantel and Lin (2000). The results over 100 frame-specific roles showed that distributional similarities get smaller error rates than Resnik and EM, with Lin's formula having the smallest error rate. Moreover, coverage of distributional similarities and Resnik are rather low. Our distributional model for selectional preferences follows her formalization.

Currently, there are several models of distributional similarity that could be used for selectional preferences. More recently, Padó and Lapata (2007) presented a study of several parameters that define a broad family of distributional similarity models, including publicly available software.

Our paper tests similar techniques to those presented above, but we evaluate selectional preference models in a setting directly related to SR classification, i.e., given a selectional preference model for a verb we find the role which fits best for a given head word. The problem is indeed qualitatively different: we do not have to choose among the head words competing for a role (as in the papers above) but among selectional preferences competing for a head word.

3 Selectional Preference Models

In this section we present all the variants for acquiring selectional preferences used in our study, and how we apply them to the SR classification.

WordNet-based SP models: we use Resnik's selectional preference model.

Distributional SP models: Given the availability of publicly available resources for distributional similarity, we used 1) a ready-made thesaurus (Lin, 1998), and 2) software (Padó and Lapata, 2007) which we run on the British National Corpus (BNC).

In the first case, Lin constructed his thesaurus based on his own similarity formula run over a large parsed corpus comprising journalism texts. The thesaurus lists, for each word, the most similar words, with their weight. In order to get the similarity for two words, we could check the entry in the thesaurus for either word. But given that the thesaurus is not symmetric, we take the average of both similarities. We will refer to this similarity measure as sim^{th}_{lin}. Another option is to use second-order similarity, where we compute the similarity of two words using the entries in the thesaurus, either using the cosine or Jaccard measures. We will refer to these similarity measures as sim^{th2}_{jac} and sim^{th2}_{cos} hereinafter.

For the second case, we tried the optimal parameters as described in (Padó and Lapata, 2007, p. 179): word-based space, medium context, log-likelihood association, and 2,000 basis elements. We tested Jaccard, cosine and Lin's measure (Lin, 1998) for similarity, yielding sim_{jac}, sim_{cos} and sim_{lin}, respectively.

3.1 Role Classification with SP Models

Given a target sentence where a predicate and several potential argument and adjunct head words occur, the goal is to assign a role label to each of the head words. The classification of candidate head words is performed independently of each other.

Since we want to evaluate the ability of selectional preference models to discriminate among different roles, this is the only knowledge that will be used to perform classification (avoiding the inclusion of any other feature commonly used in SRL). Thus, for each head word, we will simply select the role (r) of the predicate (p) which fits best the head word (w). This selection rule is formalized as:

$R(p, w) = \arg\max_{r \in Roles(p)} S(p, r, w)$

being $S(p, r, w)$ the prediction of the selectional preference model, which can be instantiated with all the variants mentioned above.

For the sake of comparison we also define a lexical baseline model, which will determine the contribution of lexical features in argument classification. For a test pair (p, w) the model returns the role under which the head word occurred most often in the training data given the predicate.

4 Experimental Setting

The data used in this work is the benchmark corpus provided by the CoNLL-2005 shared task on SRL (Carreras and Màrquez, 2005). The dataset, of over 1 million tokens, comprises PropBank sections 02-21 for training, and sections 24 and 23 for development and test, respectively. In these experiments, NEG, DIS and MOD arguments have been discarded because, apart from not being considered "pure" adjunct roles, the selectional preferences implemented in this study are not able to deal with non-nominal argument heads.

The predicate–rol–head (p, r, w) triples for generalizing the selectional preferences are extracted from the arguments of the training set, yielding 71,240 triples, from which 5,587 different predicate-role selectional preferences (p, r) are derived by instantiating the different models in Section 3.

Selectional preferences are then used, to predict the corresponding roles of the (p, w) pairs from the test corpora. The test set contains 4,134 pairs (covering 505 different predicates) to be classified into the appropriate role label. In order to study the behavior on out-of-domain data, we also tested on the PropBanked part of the Brown corpus. This corpus contains 2,932 (p, w) pairs covering 491 different predicates.

The performance of each selectional preference model is evaluated by calculating the standard *precision*, *recall* and F_1 measures. It is worth mentioning that none of the models is able to predict the role when facing an unknown head word. This happens more often with WordNet based models, which have a lower word coverage compared to distributional similarity–based models.

5 Results and Discussion

The results are presented in Table 1. The lexical row corresponds to the baseline lexical match method. The following row corresponds to the WordNet-based selectional preference model. The distributional models follow, including the results obtained by the three similarity formulas on the co-occurrences extracted from the BNC (sim_{Jac}, sim_{cos} sim_{Lin}), and the results obtained when using Lin's thesaurus directly (sim^{th}_{Lin}) and as a second-order vector (sim^{th2}_{Jac} and sim^{th2}_{cos}).

	prec.	rec.	F_1	prec.	recall	F_1
$lexical$	**.779**	.349	.482	**.663**	.059	.108
res	.589	.495	.537	.505	.379	.433
sim_{Jac}	.573	.564	.569	.481	.452	.466
sim_{cos}	.607	.598	.602	.507	.476	.491
sim_{Lin}	.580	.560	.570	.500	.470	.485
sim^{th}_{Lin}	.635	.625	.630	.494	.464	.478
sim^{th2}_{Jac}	.657	**.646**	**.651**	.531	**.499**	**.515**
sim^{th2}_{cos}	.654	.644	.649	.531	**.499**	**.515**

Table 1: Results for WSJ test (left), and Brown test (right)

As expected, the lexical baseline attains very high precision in all datasets, which underscores the importance of the lexical head word features in argument classification. The recall is quite low, specially in Brown, confirming and extending (Pradhan et al., 2008), which also reports similar performance drops when doing argument classification on out-of-domain data.

One of the main goals of our experiments is to overcome the data sparseness of lexical features both on in-domain and out-of-domain data. All our selectional preference models improve over the lexical matching baseline in recall, up to 30 absolute percentage points in the WSJ test dataset and 44 absolute percentage points in the Brown corpus. This comes at the cost of reduced precision, but the overall F-score shows that all selectional preference models improve over the baseline, with up to 17 absolute percentage points on the WSJ datasets and 41 absolute percentage points on the Brown dataset. The results, thus, show that selectional preferences are indeed alleviating the lexical sparseness problem.

As an example, consider the following head words of potential arguments of the verb *wear* found in the test set: *doctor*, *men*, *tie*, *shoe*. None of these nouns occurred as heads of arguments of *wear* in the training data, and thus the lexical feature would be unable to predict any role for them. Using selectional preferences, we successfully assigned the `Arg0` role to *doctor* and *men*, and the `Arg1` role to *tie* and *shoe*.

Regarding the selectional preference variants, WordNet-based and first-order distributional similarity models attain similar levels of precision, but the former are clearly worse on recall and F_1.

The performance loss on recall can be explained by the worse lexical coverage of WordNet when compared to automatically generated thesauri. Examples of words missing in WordNet include abbreviations (e.g., *Inc.*, *Corp.*) and brand names (e.g., *Texaco*, *Sony*). The second-order distributional similarity measures perform best overall, both in precision and recall. As far as we know, it is the first time that these models are applied to selectional preference modeling, and they prove to be a strong alternative to first-order models. The relative performance of the methods is consistent across the two datasets, stressing the robustness of all methods used.

Regarding the use of similarity software (Padó and Lapata, 2007) on the BNC vs. the use of Lin's ready-made thesaurus, both seem to perform similarly, as exemplified by the similar results of sim_{Lin} and sim^{th}_{Lin}. The fact that the former performed better on the Brown data, and worse on the WSJ data could be related to the different corpora used to compute the co-occurrence, balanced corpus and journalism texts respectively. This could be an indication of the potential of distributional thesauri to adapt to the target domain.

Regarding the similarity metrics, the cosine seems to perform consistently better for first-order distributional similarity, while Jaccard provided slightly better results for second-order similarity.

The best overall performance was for second-order similarity, also using the cosine. Given the computational complexity involved in building a complete thesaurus based on the similarity software, we used the ready-made thesaurus of Lin, but could not try the second-order version on BNC.

6 Conclusions and Future Work

We have empirically shown how automatically generated selectional preferences, using WordNet and distributional similarity measures, are able to effectively generalize lexical features and, thus, improve classification performance in a large-scale argument classification task on the CoNLL-2005 dataset. The experiments show substantial gains on recall and F_1 compared to lexical matching, both on the in-domain WSJ test and, especially, on the out-of-domain Brown test.

Alternative selectional models were studied and compared. WordNet-based models attain good levels of precision but lower recall than distributional similarity methods. A new second-order similarity method proposed in this paper attains the best results overall in all datasets.

The evidence gathered in this paper suggests that using semantic knowledge in the form of selectional preferences has a high potential for improving the results of a full system for SRL, specially when training data is scarce or when applied to out-of-domain corpora.

Current efforts are devoted to study the integration of the selectional preference models presented in this paper in a in-house SRL system. We are particularly interested in domain adaptation, and whether distributional similarities can profit from domain corpora for better performance.

Acknowledgments

This work has been partially funded by the EU Commission (project KYOTO ICT-2007-211423) and Spanish Research Department (project KNOW TIN2006-15049-C03-01). Beñat enjoys a PhD grant from the University of the Basque Country.

References

Carsten Brockmann and Mirella Lapata. 2003. Evaluating and combining approaches to selectional preference acquisition. In *Proceedings of the 10th Conference of the European Chapter of the ACL*, pages 27–34.

X. Carreras and L. Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, MI, USA.

Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 216–223, Prague, Czech Republic.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL*, pages 768–774.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199, June.

Patrick Pantel and Dekang Lin. 2000. An unsupervised approach to prepositional phrase attachment using contextually similar words. In *Proceedings of the 38th Annual Conference of the ACL*, pages 101–108.

S. Pradhan, W. Ward, and J. H. Martin. 2008. Towards robust semantic role labeling. *Computational Linguistics*, 34(2).

Philip Resnik. 1993. Semantic classes and syntactic ambiguity. In *Proceedings of the workshop on Human Language Technology*, pages 278–283, Morristown, NJ, USA.

A Syntactic and Lexical-Based Discourse Segmenter

Milan Tofiloski
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
mta45@sfu.ca

Julian Brooke
Department of Linguistics
Simon Fraser University
Burnaby, BC, Canada
jab18@sfu.ca

Maite Taboada
Department of Linguistics
Simon Fraser University
Burnaby, BC, Canada
mtaboada@sfu.ca

Abstract

We present a syntactic and lexically based discourse segmenter (SLSeg) that is designed to avoid the common problem of over-segmenting text. Segmentation is the first step in a discourse parser, a system that constructs discourse trees from elementary discourse units. We compare SLSeg to a probabilistic segmenter, showing that a conservative approach increases precision at the expense of recall, while retaining a high F-score across both formal and informal texts.

1 Introduction*

Discourse segmentation is the process of decomposing discourse into elementary discourse units (EDUs), which may be simple sentences or clauses in a complex sentence, and from which discourse trees are constructed. In this sense, we are performing low-level discourse segmentation, as opposed to segmenting text into chunks or topics (e.g., Passonneau and Litman (1997)). Since segmentation is the first stage of discourse parsing, quality discourse segments are critical to building quality discourse representations (Soricut and Marcu, 2003). Our objective is to construct a discourse segmenter that is robust in handling both formal (newswire) and informal (online reviews) texts, while minimizing the insertion of incorrect discourse boundaries. Robustness is achieved by constructing discourse segments in a principled way using syntactic and lexical information.

Our approach employs a set of rules for inserting segment boundaries based on the syntax of each sentence. The segment boundaries are then further refined by using lexical information that takes into consideration lexical cues, including multi-word expressions. We also identify clauses that are parsed as discourse segments, but are not in fact independent discourse units, and join them to the matrix clause.

Most parsers can break down a sentence into constituent clauses, approaching the type of output that we need as input to a discourse parser. The segments produced by a parser, however, are too fine-grained for discourse purposes, breaking off complement and other clauses that are not in a discourse relation to any other segment. For this reason, we have implemented our own segmenter, utilizing the output of a standard parser. The purpose of this paper is to describe our syntactic and lexical-based segmenter (SLSeg), demonstrate its performance against state-of-the-art systems, and make it available to the wider community.

2 Related Work

Soricut and Marcu (2003) construct a statistical discourse segmenter as part of their sentence-level discourse parser (SPADE), the only implementation available for our comparison. SPADE is trained on the RST Discourse Treebank (Carlson et al., 2002). The probabilities for segment boundary insertion are learned using lexical and syntactic features. Subba and Di Eugenio (2007) use neural networks trained on RST-DT for discourse segmentation. They obtain an F-score of 84.41% (86.07% using a perfect parse), whereas SPADE achieved 83.1% and 84.7% respectively.

Thanh et al. (2004) construct a rule-based segmenter, employing manually annotated parses from the Penn Treebank. Our approach is conceptually similar, but we are only concerned with established discourse relations, i.e., we avoid potential *same-unit* relations by preserving NP constituency.

*This work was supported by an NSERC Discovery Grant (261104-2008) to Maite Taboada. We thank Angela Cooper and Morgan Mameni for their help with the reliability study.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 77–80,
Suntec, Singapore, 4 August 2009.

3 Principles For Discourse Segmentation

Our primary concern is to capture interesting discourse relations, rather than all possible relations, i.e., capturing more specific relations such as Condition, Evidence or Purpose, rather than more general and less informative relations such as Elaboration or Joint, as defined in Rhetorical Structure Theory (Mann and Thompson, 1988). By having a stricter definition of an elementary discourse unit (EDU), this approach increases precision at the expense of recall.

Grammatical units that are candidates for discourse segments are clauses and sentences. Our basic principles for discourse segmentation follow the proposals in RST as to what a minimal unit of text is. Many of our differences with Carlson and Marcu (2001), who defined EDUs for the RST Discourse Treebank (Carlson et al., 2002), are due to the fact that we adhere closer to the original RST proposals (Mann and Thompson, 1988), which defined as 'spans' adjunct clauses, rather than complement (subject and object) clauses. In particular, we propose that complements of attributive and cognitive verbs (*He said (that)...*, *I think (that)...*) are not EDUs. We preserve consistency by not breaking at direct speech (*"X," he said.*). Reported and direct speech are certainly important in discourse (Prasad et al., 2006); we do not believe, however, that they enter discourse relations of the type that RST attempts to capture.

In general, adjunct, but not complement clauses are discourse units. We require all discourse segments to contain a verb. Whenever a discourse boundary is inserted, the two newly created segments must each contain a verb. We segment coordinated clauses (but not coordinated VPs), adjunct clauses with either finite or non-finite verbs, and non-restrictive relative clauses (marked by commas). In all cases, the choice is motivated by whether a discourse relation could hold between the resulting segments.

4 Implementation

The core of the implementation involves the construction of 12 syntactically-based segmentation rules, along with a few lexical rules involving a list of stop phrases, discourse cue phrases and word-level parts of speech (POS) tags. First, paragraph boundaries and sentence boundaries using NIST's sentence segmenter[1] are inserted. Second, a statistical parser applies POS tags and the sentence's syntactic tree is constructed. Our syntactic rules are executed at this stage. Finally, lexical rules, as well as rules that consider the parts-of-speech for individual words, are applied. Segment boundaries are removed from phrases with a syntactic structure resembling independent clauses that actually are used idiomatically, such as *as it stands* or *if you will*. A list of phrasal discourse cues (e.g., *as soon as, in order to*) are used to insert boundaries not derivable from the parser's output (phrases that begin with *in order to...* are tagged as PP rather than SBAR). Segmentation is also performed within parentheticals (marked by parentheses or hyphens).

5 Data and Evaluation

5.1 Data

The gold standard test set consists of 9 human-annotated texts. The 9 documents include 3 texts from the RST literature[2], 3 online product reviews from Epinions.com, and 3 Wall Street Journal articles taken from the Penn Treebank. The texts average 21.2 sentences, with the longest text having 43 sentences and the shortest having 6 sentences, for a total of 191 sentences and 340 discourse segments in the 9 gold-standard texts.

The texts were segmented by one of the authors following guidelines that were established from the project's beginning and was used as the gold standard. The annotator was not directly involved in the coding of the segmenter. To ensure the guidelines followed clear and sound principles, a reliability study was performed. The guidelines were given to two annotators, both graduate students in Linguistics, that had no direct knowledge of the project. They were asked to segment the 9 texts used in the evaluation.

Inter-annotator agreement across all three annotators using Kappa was .85, showing a high level of agreement. Using F-score, average agreement of the two annotators against the gold standard was also high at .86. The few disagreements were primarily due to a lack of full understanding of the guidelines (e.g., the guidelines specify to break adjunct clauses when they contain a verb, but one of the annotators segmented prepositional phrases

[1] http://duc.nist.gov/duc2004/software/duc2003.breakSent.tar.gz

[2] Available from the RST website http://www.sfu.ca/rst/

	Epinions			Treebank			Original RST			Combined Total		
System	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
Baseline	.22	.70	.33	.27	.89	.41	.26	**.90**	.41	.25	**.80**	.38
SPADE (coarse)	.59	.66	.63	.63	**1.0**	.77	.64	.76	.69	.61	.79	.69
SPADE (original)	.36	.67	.46	.37	**1.0**	.54	.38	.76	.50	.37	.77	.50
Sundance	.54	.56	.55	.53	.67	.59	.71	.47	.57	.56	.58	.57
SLSeg (Charniak)	**.97**	.66	**.79**	**.89**	.86	**.87**	**.94**	.76	**.84**	**.93**	.74	**.83**
SLSeg (Stanford)	.82	**.74**	.77	.82	.86	.84	.88	.71	.79	.83	.77	.80

Table 1: Comparison of segmenters

that had a similar function to a full clause). With high inter-annotator agreement (and with any disagreements and errors resolved), we proceeded to use the co-author's segmentations as the gold standard.

5.2 Evaluation

The evaluation uses standard precision, recall and F-score to compute correctly inserted segment boundaries (we do not consider sentence boundaries since that would inflate the scores). Precision is the number of boundaries in agreement with the gold standard. Recall is the total number of boundaries correct in the system's output divided by the number of total boundaries in the gold standard.

We compare the output of SLSeg to SPADE. Since SPADE is trained on RST-DT, it inserts segment boundaries that are different from what our annotation guidelines prescribe. To provide a fair comparison, we implement a coarse version of SPADE where segment boundaries prescribed by the RST-DT guidelines, but not part of our segmentation guidelines, are manually removed. This version leads to increased precision while maintaining identical recall, thus improving F-score.

In addition to SPADE, we also used the Sundance parser (Riloff and Phillips, 2004) in our evaluation. Sundance is a shallow parser which provides clause segmentation on top of a basic word-tagging and phrase-chunking system. Since Sundance clauses are also too fine-grained for our purposes, we use a few simple rules to collapse clauses that are unlikely to meet our definition of EDU. The baseline segmenter in Table 1 inserts segment boundaries before and after all instances of S, SBAR, SQ, SINV, SBARQ from the syntactic parse (text spans that represent full clauses able to stand alone as sentential units). Finally, two parsers are compared for their effect on segmentation quality: Charniak (Charniak, 2000) and Stanford (Klein and Manning, 2003).

5.3 Qualitative Comparison

Comparing the outputs of SLSeg and SPADE on the Epinions.com texts illustrates key differences between the two approaches.

> [Luckily we bought the extended protection plans from Lowe's,] # [so we are waiting] [for Whirlpool to decide] [if they want to do the costly repair] [or provide us with a new machine].

In this example, SLSeg inserts a single boundary (#) before the word *so*, whereas SPADE inserts four boundaries (indicated by square brackets). Our breaks err on the side of preserving semantic coherence, e.g., the segment *for Whirlpool to decide* depends crucially on the adjacent segments for its meaning. In our opinion, the relations between these segments are properly the domain of a semantic, but not a discourse, parser. A clearer example that illustrates the pitfalls of fine-grained discourse segmenting is shown in the following output from SPADE:

> [The thing] [that caught my attention was the fact] [that these fantasy novels were marketed...]

Because the segments are a restrictive relative clause and a complement clause, respectively, SLSeg does not insert any segment boundaries.

6 Results

Results are shown in Table 1. The combined informal and formal texts show SLSeg (using Charniak's parser) with high precision; however, our overall recall was lower than both SPADE and the baseline. The performance of SLSeg on the informal and formal texts is similar to our perfor-

mance overall: high precision, nearly identical recall. Our system outperforms all the other systems in both precision and F-score, confirming our hypothesis that adapting an existing system would not provide the high-quality discourse segments we require.

The results of using the Stanford parser as an alternative to the Charniak parser show that the performance of our system is parser-independent. High F-score in the Treebank data can be attributed to the parsers having been trained on Treebank. Since SPADE also utilizes the Charniak parser, the results are comparable.

Additionally, we compared SLSeg and SPADE to the original RST segmentations of the three RST texts taken from RST literature. Performance was similar to that of our own annotations, with SLSeg achieving an F-score of .79, and SPADE attaining .38. This demonstrates that our approach to segmentation is more consistent with the original RST guidelines.

7 Discussion

We have shown that SLSeg, a conservative rule-based segmenter that inserts fewer discourse boundaries, leads to higher precision compared to a statistical segmenter. This higher precision does not come at the expense of a significant loss in recall, as evidenced by a higher F-score. Unlike statistical parsers, our system requires no training when porting to a new domain.

All software and data are available[3]. The discourse-related data includes: a list of clause-like phrases that are in fact discourse markers (e.g., *if you will, mind you*); a list of verbs used in *to*-infinitival and *if* complement clauses that should not be treated as separate discourse segments (e.g., *decide* in *I decided to leave the car at home*); a list of unambiguous lexical cues for segment boundary insertion; and a list of attributive/cognitive verbs (e.g., *think, said*) used to prevent segmentation of floating attributive clauses.

Future work involves studying the robustness of our discourse segments on other corpora, such as formal texts from the medical domain and other informal texts. Also to be investigated is a quantitative study of the effects of high-precision/low-recall vs. low-precision/high-recall segmenters on the construction of discourse trees. Besides its use in automatic discourse parsing, the system could assist manual annotators by providing a set of discourse segments as starting point for manual annotation of discourse relations.

[3] http://www.sfu.ca/~mtaboada/research/SLSeg.html

References

Lynn Carlson and Daniel Marcu. 2001. *Discourse Tagging Reference Manual*. ISI Technical Report ISI-TR-545.

Lynn Carlson, Daniel Marcu and Mary E. Okurowski. 2002. *RST Discourse Treebank*. Philadelphia, PA: Linguistic Data Consortium.

Eugene Charniak. 2000. A Maximum-Entropy Inspired Parser. *Proc. of NAACL*, pp. 132–139. Seattle, WA.

Barbara J. Grosz and Candace L. Sidner. 1986. Attention, Intentions, and the Structure of Discourse. *Computational Linguistics*, 12:175–204.

Dan Klein and Christopher D. Manning. 2003. Fast Exact Inference with a Factored Model for Natural Language Parsing. *Advances in NIPS 15 (NIPS 2002)*, Cambridge, MA: MIT Press, pp. 3–10.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8:243–281.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, MA.

Rebecca J. Passonneau and Diane J. Litman. 1997. Discourse Segmentation by Human and Automated Means. *Computational Linguistics*, 23(1):103–139.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Aravind Joshi and Bonnie Webber. 2006. Attribution and its Annotation in the Penn Discourse TreeBank. *Traitement Automatique des Langues*, 47(2):43–63.

Ellen Riloff and William Phillips. 2004. *An Introduction to the Sundance and AutoSlog Systems*. University of Utah Technical Report #UUCS-04-015.

Radu Soricut and Daniel Marcu. 2003. Sentence Level Discourse Parsing Using Syntactic and Lexical Information. *Proc. of HLT-NAACL*, pp. 149–156. Edmonton, Canada.

Rajen Subba and Barbara Di Eugenio. 2007. Automatic Discourse Segmentation Using Neural Networks. *Proc. of the 11th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 189–190. Rovereto, Italy.

Huong Le Thanh, Geetha Abeysinghe, and Christian Huyck. 2004. Automated Discourse Segmentation by Syntactic Information and Cue Phrases. *Proc. of IASTED*. Innsbruck, Austria.

Realistic Grammar Error Simulation using Markov Logic

Sungjin Lee
Pohang University of Science and Technology
Pohang, Korea
junion@postech.ac.kr

Gary Geunbae Lee
Pohang University of Science and Technology
Pohang, Korea
gblee@postech.ac.kr

Abstract

The development of Dialog-Based Computer-Assisted Language Learning (DB-CALL) systems requires research on the simulation of language learners. This paper presents a new method for generation of grammar errors, an important part of the language learner simulator. Realistic errors are generated via Markov Logic, which provides an effective way to merge a statistical approach with expert knowledge about the grammar error characteristics of language learners. Results suggest that the distribution of simulated grammar errors generated by the proposed model is similar to that of real learners. Human judges also gave consistently close judgments on the quality of the real and simulated grammar errors.

1 Introduction

Second Language Acquisition (SLA) researchers have claimed that feedback provided during conversational interaction facilitates the acquisition process. Thus, interest in developing Dialog-Based Computer Assisted Language Learning (DB-CALL) systems is rapidly increasing. However, developing DB-CALL systems takes a long time and entails a high cost in collecting learners' data. Also, evaluating the systems is not a trivial task because it requires numerous language learners with a wide range of proficiency levels as subjects.

While previous studies have considered user simulation in the development and evaluation of spoken dialog systems (Schatzmann et al., 2006), they have not yet simulated grammar errors because those systems were assumed to be used by native speakers, who normally produce few grammar errors in utterances. However, as telephone-based information access systems become more commonly available to the general public, the inability to deal with non-native speakers is becoming a serious limitation since, at least for some applications, (e.g. tourist information, legal/social advice) non-native speakers represent a significant portion of the everyday user population. Thus, (Raux and Eskenazi, 2004) conducted a study on adaptation of spoken dialog systems to non-native users. In particular, DB-CALL systems should obviously deal with grammar errors because language learners naturally commit numerous grammar errors. Thus grammar error simulation should be embedded in the user simulation for the development and evaluation of such systems.

In Foster's (2007) pioneering work, she described a procedure which automatically introduces frequently occurring grammatical errors into sentences to make ungrammatical training data for a robust parser. However the algorithm cannot be directly applied to grammar error generation for language learner simulation for several reasons. First, it either introduces one error per sentence or none, regardless of how many words of the sentence are likely to generate errors. Second, it determines which type of error it will create only by relying on the relative frequencies of error types and their relevant parts of speech. This, however, can result in unrealistic errors. As exemplified in Table 1, when the algorithm tries to create an error by deleting a word, it would probably omit the word 'go' because verb is one of the most frequent parts of speech omitted resulting in an unrealistic error like the first simulated output. However, Korean/Japanese language learners of English tend to make subject-verb agreement errors, omission errors of the preposition of prepositional verbs, and omission errors of articles because their first language does not have similar grammar rules so that they may be slow on the uptake of such constructs. Thus, they often commit errors like the second simulated output.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 81–84,
Suntec, Singapore, 4 August 2009.

Input sentence
He wants to go to a movie theater
Unrealistic simulated output
He wants to to a movie theater
Realistic simulated output
He want go to movie theater

Table 1: Examples of simulated outputs

This paper develops an approach to statistical grammar error simulation that can incorporate this type of knowledge about language learners' error characteristics and shows that it does indeed result in realistic grammar errors. The approach is based on Markov logic, a representation language that combines probabilistic graphical models and first-order logic (Richardson and Domingos, 2006). Markov logic enables concise specification of very complex models. Efficient open-source Markov logic learning and inference algorithms were used to implement our solution.

We begin by describing the overall process of grammar error simulation and then briefly reviewing the necessary background in Markov logic. We then describe our Markov Logic Network (MLN) for grammar error simulation. Finally, we present our experiments and results.

2 Overall process of grammar error simulation

The task of grammar error simulation is to generate an ill-formed sentence when given a well-formed input sentence. The generation procedure involves three steps: 1) Generating probability over error types for each word of the well-formed input sentence through MLN inference 2) Determining an error type by sampling the generated probability for each word 3) Creating an ill-formed output sentence by realizing the chosen error types (Figure 1).

3 Markov Logic

Markov logic is a probabilistic extension of finite first-order logic (Richardson and Domingos, 2006). An MLN is a set of weighted first-order clauses. Together with a set of constants, it defines a Markov network with one node per ground atom and one feature per ground clause. The weight of a feature is the weight of the first-order clause that originated it. The probability of a state x in such a network is given by $P(x) = (1/Z)\ exp\ (\sum_i w_i f_i(x))$, where Z is a normalization constant, w_i is the weight of the ith clause, $f_i = 1$ if the ith clause is true, and $f_i = 0$ otherwise.

Markov logic makes it possible to compactly specify probability distributions over complex relational domains. We used the learning and inference algorithms provided in the open-source Alchemy package (Kok et al., 2006). In particular, we performed inference using the belief propagation algorithm (Pearl, 1988), and generative weight learning.

4 An MLN for Grammar Error Simulation

This section presents our MLN implementation which consists of three components: 1) Basic formulas based on parts of speech, which are comparable to Foster's method 2) Analytic formulas drawn from expert knowledge obtained by error analysis on a learner corpus 3) Error limiting formulas that penalize statistical model's over-generation of nonsense errors.

4.1 Basic formulas

Error patterns obtained by error analysis, which might capture a lack or an over-generalization of knowledge of a particular construction, cannot explain every error that learners commit. Because an error can take the form of a performance slip which can randomly occur due to carelessness or tiredness, more general formulas are needed as a default case. The basic formulas are represented by the simple rule:

• $PosTag(s, i, +pt) \Rightarrow ErrorType(s, i, +et)$

where all free variables are implicitly universally quantified. The "$+pt, +et$" notation signifies that the MLN contains an instance of this rule for each (part of speech, error type) pair. The evi-

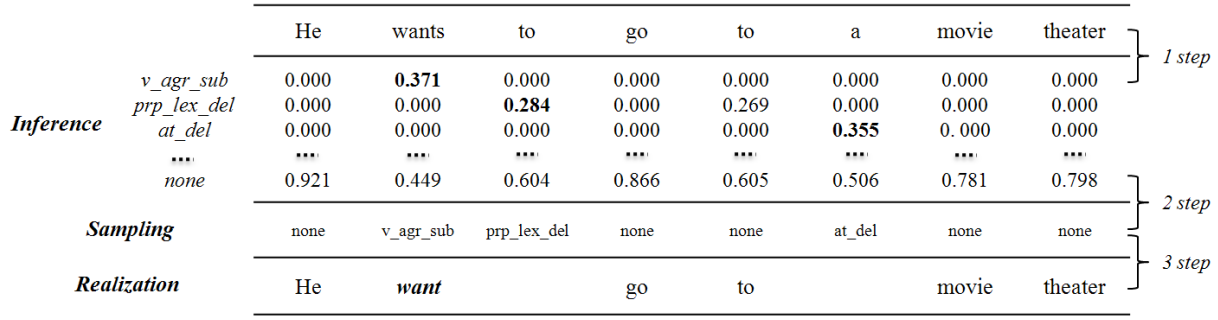


Figure 1: An example process of grammar error simulation

dence predicate in this case is $PosTag(s, i, pt)$, which is true iff the ith position of the sentence s has the part of speech pt. The query predicate is $ErrorType(s, i, et)$. It is true iff the ith position of the sentence s has the error type et, and inferring it returns the probability that the word at position i would commit an error of type et.

4.2 Analytic formulas

On top of the basic formulas, analytic formulas add concrete knowledge of realistic error characteristics of language learners. Error analysis and linguistic differences between the first language and the second language can identify various error sources for each error type. We roughly categorize the error sources into three groups for explanation: 1) Over-generalization of the rules of the second language 2) Lack of knowledge of some rules of the second language 3) Applying rules and forms of the first language into the second language.

Often, English learners commit pluralization error with irregular nouns. This is because they over-generalize the pluralization rule, i.e. attaching 's/es', so that they apply the rule even to irregular nouns such as 'fish' and 'feet' etc. This characteristic is captured by the simple formula:

• $IrregularPluralNoun(s, i) \wedge PosTag(s, i, NNS)$
$\Rightarrow ErrorType(s, i, N\_NUM\_SUB)$

where $IrregularPluralNoun(s, i)$ is true iff the ith word of the sentence s is an irregular plural and N_NUM_SUB is the abbreviation for substitution by noun number error.

One trivial error caused by a lack of knowledge of the second language is using the singular noun form for weekly events:

• $Word(s, i-1, on) \wedge DayNoun(s, i)$
$\wedge PosTag(s, i, NNS) \Rightarrow ErrorType(s, i, N\_NUM\_SUB)$

where $Word(s, i-1, on)$ is true iff the $i-1$th word is 'on' and $DayNoun(s, i)$ is true iff the ith word of the sentence s is a noun describing day like Sunday(s). Another example is use of plurals behind 'every' due to the ignorance that a noun modified by 'every' should be singular:

• $Word(s, di, every) \wedge DeterminerRel(s, di, ni)$
$\Rightarrow ErrorType(s, ni, N\_NUM\_SUB)$

where $DeterminerRel(s, di, ni)$ is true iff the dith word is the determiner of the nith word.

An example of errors by applying the rules of the first language is that Korean/Japanese often allows omission of the subject of a sentence; thus, they easily commit the subject omission error. The following formula is for the case:

• $Subject(s, i) \Rightarrow ErrorType(s, i, N\_LXC\_DEL)$

where $Subject(s, i)$ is true iff the ith word is the subject and N_LXC_DEL is the abbreviation for deletion by noun lexis error.[1]

4.3 Error limiting formulas

A number of elementary formulas explicitly stated as hard formulas prevent the MLN from generating improbable errors that might result from over-generations of the statistical model. For example, a verb complement error should not have a probability at the words that are not complements of a verb:

• $!VerbComplement(s, vi, ci)$
$\Rightarrow !ErrorType(s, ci, V\_CMP\_SUB).$

where "!" denotes logically 'not' and "." at the end signifies that it is a hard formula. Hard formulas are given maximum weight during inference. $VerbComplement(s, vi, ci)$ is true iff the cith word is a complement of the verb at the vith position and V_CMP_SUB is the abbreviation for substitution by verb complement error.

5 Experiments

Experiments used the NICT JLE Corpus, which is speech samples from an English oral proficiency interview test, the ACTFL-ALC Standard Speaking Test (SST). 167 of the files are error annotated. The error tagset consists of 47 tags that are described in Izumi (2005). We appended structural type of errors (substitution, addition, deletion) to the original error types because structural type should be determined when creating an error. For example, V_TNS_SUB consists of the original error type V_TNS (verb tense) and structural type SUB (substitution). Level-specific language learner simulation was accomplished by dividing the 167 error annotated files into 3 level groups: Beginner(level1-4), Intermediate(level5-6), Advanced(level7-9).

The grammar error simulation was compared with real learners' errors and the baseline model using only basic formulas comparable to Foster's algorithm, with 10-fold cross validations performed for each group. The validation results were added together across the rounds to compare the number of simulated errors with the number of real errors. Error types that occurred less than 20 times were excluded to improve reliability. Result graphs suggest that the distribution of simulated grammar errors generated by the proposed model using all formulas is similar to that of real learners for all level groups and the

[1] Because space is limited, all formulas can be found at http://isoft.postech.ac.kr/ges/grm_err_sim.mln

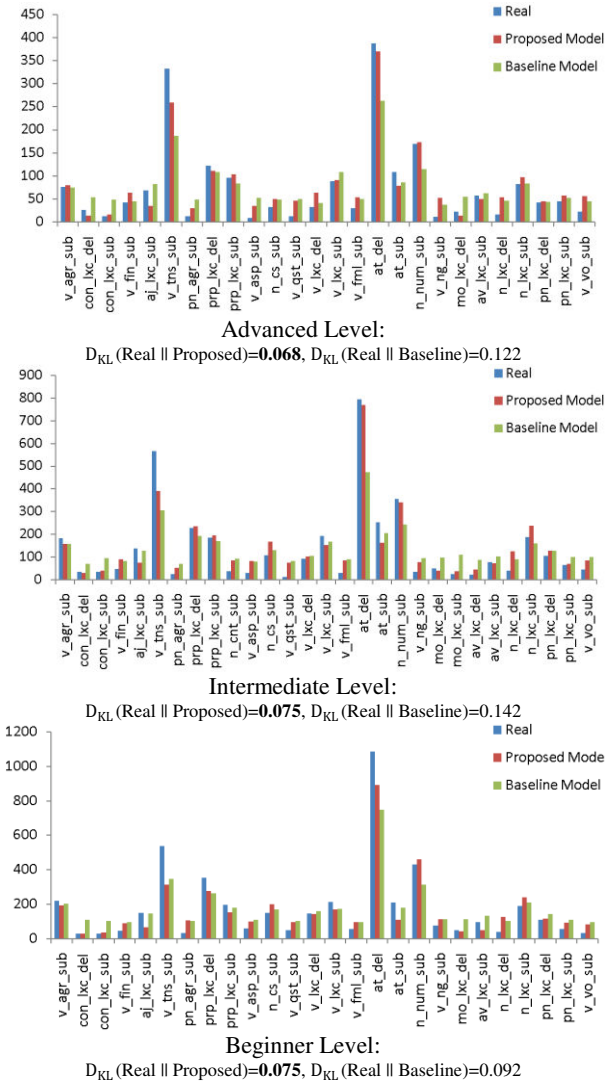


Figure 2: Comparison between the distributions of the real and simulated data

proposed model outperforms the baseline model using only the basic formulas. The Kullback-Leibler divergences, a measure of the difference between two probability distributions, were also measured for quantitative comparison. For all level groups, the Kullback-Leibler divergence of the proposed model from the real is less than that of the baseline model (Figure 2).

Two human judges verified the overall realism of the simulated errors. They evaluated 100 randomly chosen sentences consisting of 50 sentences each from the real and simulated data. The sequence of the test sentences was mixed so that the human judges did not know whether the source of the sentence was real or simulated. They evaluated sentences with a two-level scale (0: Unrealistic, 1: Realistic). The result shows that the inter evaluator agreement (kappa) is moderate and that both judges gave relatively close judgments on the quality of the real and simulated data (Table 2).

	Human 1	Human 2	Average	Kappa
Real	0.84	0.8	0.82	0.46
Simulated	0.8	0.8	0.8	0.5

Table 2: Human evaluation results

6 Summary and Future Work

This paper introduced a somewhat new research topic, grammar error simulation. Expert knowledge of error characteristics was imported to statistical modeling using Markov logic, which provides a theoretically sound way of encoding knowledge into probabilistic first order logic. Results indicate that our method can make an error distribution more similar to the real error distribution than the baseline and that the quality of simulated sentences is relatively close to that of real sentences in the judgment of human evaluators. Our future work includes adding more expert knowledge through error analysis to incrementally improve the performance. Furthermore, actual development and evaluation of a DB-CALL system will be arranged so that we may investigate how much the cost of collecting data and evaluation would be reduced by using language learner simulation.

Acknowledgement

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute for Information Technology Advancement) (IITA-2009-C1090-0902-0045).

References

Foster, J. 2007. Treebanks Gone Bad: Parser evaluation and retraining using a treebank of ungrammatical sentences. IJDAR, 10(3-4), 129-145.

Izumi, E et al. 2005. Error Annotation for Corpus of Japanese Learner English. In Proc. International Workshop on Linguistically Interpreted Corpora

Kok, S. et al. 2006. The Alchemy system for statistical relational AI. http://alchemy.cs.washington.edu/.

Pearl, J. 1988. Probabilistic Reasoning in Intelligent Systems Morgan Kaufmann.

Raux, A. and Eskenazi, M. 2004. Non-Native Users in the Let's Go!! Spoken Dialogue System: Dealing with Linguistic Mismatch, HLT/NAACL.

Richardson, M. and Domingos, P. 2006. Markov logic networks. Machine Learning, 62(1):107-136.

Schatzmann, J. et al. 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies, The Knowledge Engineering Review, Vol. 21:2, 97–126

Discriminative Approach to Predicate-Argument Structure Analysis with Zero-Anaphora Resolution

Kenji Imamura, Kuniko Saito, and **Tomoko Izumi**
NTT Cyber Space Laboratories, NTT Corporation
1-1 Hikarinooka, Yokosuka, Kanagawa, 239-0847, Japan
{imamura.kenji,saito.kuniko,izumi.tomoko}@lab.ntt.co.jp

Abstract

This paper presents a predicate-argument structure analysis that simultaneously conducts zero-anaphora resolution. By adding noun phrases as candidate arguments that are not only in the sentence of the target predicate but also outside of the sentence, our analyzer identifies arguments regardless of whether they appear in the sentence or not. Because we adopt discriminative models based on maximum entropy for argument identification, we can easily add new features. We add language model scores as well as contextual features. We also use contextual information to restrict candidate arguments.

1 Introduction

Predicate-argument structure analysis is a type of semantic role labeling, which is an important module to extract event information such as "*who did what to whom*" from a sentence. There are many arguments called *zero pronouns* that do not appear in the surface of a sentence in Japanese. In this case, predicate-argument structures cannot be constructed if we only rely on the syntactic information of a single sentence. Similar phenomena also happen in English noun predicates, in which arguments of noun predicates sometimes do not exist in the sentence due to things such as ellipses (Jiang and Ng, 2006). To correctly extract the structures from such sentences, it is necessary to resolve what zero pronouns refer to by using other information such as context.

Although predicate-argument structure analysis and zero-anaphora resolution are closely related, it was not until recently that these two tasks were lumped together. Due to the developments of large annotated corpora with predicate-argument and coreference relations (e.g.,(Iida et al., 2007)) and with case frames, several works using statistical models have been proposed to solve these two tasks simultaneously (Sasano et al., 2008; Taira et al., 2008).

In this paper, we present a predicate-argument structure analysis that simultaneously resolves the anaphora of zero pronouns in Japanese, based on supervised learning. The analyzer obtains candidate arguments not only from the sentence of the target predicate but also from the previous sentences. It then identifies the most likely arguments based on discriminative models. To identify arguments that appear in the sentence and are represented by zero pronouns without distinction, the analyzer introduces the following features and techniques: the language model features of noun phrases, contextual features, and restrictions of candidate arguments.

2 Predicate-Argument Structure Analyzer

2.1 Procedure and Models

The procedure of our predicate-argument structure analyzer is as follows. The input to the analyzer is an article (multiple sentences) because our target is to identify arguments spread across sentences.

1. First, each sentence is individually analyzed and segmented into base phrases by a morphological analyzer and a base phrase chunker. In Japanese, a base phrase is usually constructed by one or more content words (such as base noun phrases) and function words (such as case particles). In addition, dependency relations among base phrases are parsed by a dependency parser. In this paper, base phrases and dependency relations are acquired from an annotated corpus (i.e., correct parses).
2. Next, predicates are extracted from the base phrases. In general, a predicate is determined

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 85–88,
Suntec, Singapore, 4 August 2009.

	Name	Note
Baseline Features	Predicate	Form and POS of the predicate
	Noun	Form and POS of the headword of the candidate phrase
	Particle	Form and POS of the particle of the candidate phrase
	Path	Dependency relation between the predicate and the candidate phrase
	Passive	Passive auxiliary verbs that the predicate contains
	PhPosit	Relative phrase position between the predicate and the candidate phrase
	SentPosit	Relative sentence position between the predicate and the candidate phrase
Additional Features (c.f., Sec. 2.2 and 2.3)	LangModel	Language model scores
	Used	Flag whether the candidate phrase was used as arguments of previous predicates
	SRLOrder	Order in Salient Referent List

Table 1: Features Used in this Paper

based on parts of speech such as verbs and adjectives. In this paper, the predicates are also provided from an annotated corpus.

3. Concurrently, noun phrases and their headwords are extracted as candidate arguments from base phrases. If an argument of a predicate is a zero pronoun, it is likely that the argument itself has appeared in previous sentences. Therefore, the analyzer collects not only all phrases in the sentence but also some phrases in the previous sentences. We also add the special noun phrase NULL, which denotes that the argument of the predicate is not required or did not appear in the article (i.e., exophoric).

4. Next, features needed for an argument identifier are extracted from each pair of a predicate and a candidate argument. Features used in this paper are shown in Table 1. Baseline features are roughly those of the predicate, the noun phrase, and their relations (on the phrasal/sentential sequence and the dependency tree). For binary features, we use all combinations of these features listed above.

5. Finally, the argument identifier selects the best phrases for nominative, accusative, and dative cases from the candidate arguments (Figure 1).

In this paper, we use maximum entropy models normalized for each predicate to each case. That is, the identifier directly selects the best phrase that

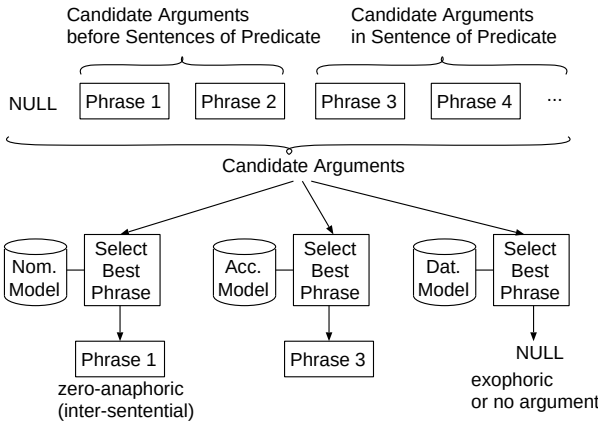


Figure 1: Summary of Argument Identification

satisfies the following equations from the candidate arguments:

$$\hat{n} = \underset{n_j \in N}{\operatorname{argmax}} P(d(n_j) = 1 | X_j; M_c) \quad (1)$$

$$P(d(n_j) = 1 | X_j; M_c) = \frac{1}{Z_c(X)} \exp \sum_k \{\lambda_{c_k} f_k(d(n_j) = 1, X_j)\} \quad (2)$$

$$Z_c(X) = \sum_{n_j \in N} \exp \sum_k \{\lambda_{c_k} f_k(d(n_j) = 1, X_j)\} \quad (3)$$

$$X_j = \langle n_j, v, A \rangle \quad (4)$$

where n, c, and v denote a noun phrase of an argument, the case, and the target predicate, respectively, N denotes a set of candidate arguments, $d(n)$ is a function that returns 1 iff the phrase n becomes the argument, and M_c denotes the model of the case c. In addition, $f_k(d(n_j) = 1, X_j)$ is a feature function, λ_{c_k} denotes a weight parameter of the feature function, and A denotes an article in which all sentences are parsed.

As shown, our analyzer can assign the best noun phrases to arguments regardless of whether they appear in the sentence or not by collecting candidates spread across multiple sentences. Furthermore, because the identifier is regarded as a selector based on the discriminative models, our analyzer has two properties: 1) New features can be easily added. 2) The precision can be improved by restricting the candidate arguments appropriately.

When we analyze predicate-argument structures and zero-anaphora resolution, syntactic information sometimes does not help because referents of zero pronouns do not appear in the sentence of the predicate. To overcome this problem,

we introduce additional information, i.e., language model scores and contextual information.

2.2 Language Models

Even if syntactic information does not help to identify arguments, we can expect that a certain noun phrase might be the correct argument of the predicate when we put it in place of the zero pronoun and the sentence becomes meaningful. Therefore, we add language model scores as features of the identifier. Because the appearance order of argument phrases is not strongly constricted in Japanese, we construct generation models that reflect dependency relations among a predicate, its case and a noun phrase. That is, we regard generation probabilities $P(n|c, v)$ acquired from the dependency tree as the scores of language models.

The language models are built from large plain texts by using a dependency parser. First, predicates and the base phrases that directly depend on the predicates are aquired from parsed sentences. Next, case particles and headwords are extracted from the base phrases. Finally, generation probabilities are computed using maximum likelihood estimation. Good-Turing discounting and backoff smoothing are also applied. Here, it is necessary to assign generation probabilities to NULLs. Regarding the training corpus that will be described in Section 3, the NULL rates of the nominative, accusative, and dative cases were 16.7%, 59.9%, and 81.6%, respectively. We assign these rates to the backoff term $P(\text{NULL}|c)$.

Using the language models, generation probabilities of the noun phrases are computed for every case of the predicate, and features that maintain the logarithms of language model scores are added ('LangModel' features in Table 1). Thus, the values of these feature functions are real.

2.3 Usage of Context

Centering theory claims that noun phrases that have been used once tend to be used again within the same context. We adopt this claim and add two different kinds of features. One is the feature that indicates whether a candidate has been used as an argument of predicates in the preceding sentences ('Used' features). However, the Used features are affected by the accuracy of the previous analyses. Thus, we also adopt the Salience Reference List (Nariyama, 2002), which only uses explicit surface case markers or a topic marker, and added their priority order to the List as another feature ('SRLOrder' feature).

	Training	Development	Test
# of Articles	1,751	480	695
# of Sentences	24,225	4,833	9,272
# of Predicates	67,145	13,594	25,500
# of Arguments			
Nom.	56,132	11,969	21,931
Acc.	26,899	5,566	10,329
Dat.	12,332	3,147	5,944

Table 2: Corpus Statistics

Another way to adopt contextual information is to restrict the candidate arguments. When we analyzed the training corpus from the viewpoint of zero pronouns, it was found that 102.2 noun phrases on average were required as candidate arguments if we did not stipulate any restrictions. When the candidate arguments we had restricted to those that had been used as arguments of the predicate appeared in a previous *one* sentence (namely, noun phrases appeared in more than one sentence before have a chance to remain), then the number of candidate arguments significantly decreased to an average of 3.2 but they covered the 62.5% of the referents of zero pronouns.

By using these characteristics, our analyzer restricts the candidate arguments to those that are of the same sentence, and those that were used as the arguments of another predicate in a previous sentence.

3 Experiments

3.1 Experimental Settings

Corpora: We used the NAIST Text Corpus version 1.4b (Iida et al., 2007) and the Kyoto Text Corpus 4.0 as the annotated corpora. We could obtain dependency and predicate-argument structures because these corpora were annotated to almost the same newspaper articles. We divided them into training, development, and test sets as shown in Table 2.

Argument Identification Models: Maximum entropy models were trained using the training set. In these experiments, we used the Gaussian prior, and the variance was tuned using the development set. Candidate argument restrictions were applied during both training and decoding.

Language Models: Language models were trained from twelve years of newspaper articles (Mainichi Shinbun newspaper 1991-2002, about

Case	Type	# of Args.	Prec.	Rec.	F
Nom.	Dep.	14,287	85.2%	88.8%	87.0%
	Zero-Intra	4,581	58.8%	43.4%	50.0%
	Zero-Inter	3,063	47.5%	7.6%	13.1%
	Total	21,931	79.4%	68.0%	73.2%
Acc.	Dep.	9,316	95.6%	92.2%	93.9%
	Zero-Intra	742	53.7%	21.6%	30.8%
	Zero-Inter	271	25.0%	0.4%	0.7%
	Total	10,329	94.3%	84.7%	89.2%
Dat.	Dep.	5,409	91.1%	72.6%	80.8%
	Zero-Intra	396	0.0%	0.0%	0.0%
	Zero-Inter	139	0.0%	0.0%	0.0%
	Total	5,944	91.1%	66.1%	76.6%

Table 3: Results on the Test Set

5.5M sentences) using the method described in Section 2.2. However, we eliminated articles that overlap the NAIST Corpus.

Evaluation: We evaluated the precision and recall rates, and F scores, all of which were computed by comparing system output and the correct answer of each argument. We also evaluated the rate at which all arguments of a predicate were completely identified as predicate-argument accuracy.

3.2 Results

The results are shown in Table 3. This table shows accuracies of the argument identification according to each case and each dependency relation between predicates and arguments. The predicate-argument accuracy on the test set was 59.4% (15,140/25,500).

First, focusing on the F scores of the Dep. relations, which denote a predicate and an argument in the same sentence and directly depend upon each other, scores of over 80% were obtained for all cases. Compared with Taira et al. (2008), they were higher in the nominative and accusative cases but were lower in the dative case. Overall, we obtained F scores between 73.2% and 89.2%.

Next, focusing on the intra-sentential (Zero-Intra) and inter-sentential (Zero-Intra) zero-anaphora, the analyzer identified arguments at some level from the viewpoint of precision. However, the recall rates and F scores were very low. The Zero-Inter recall rate for the nominative case, in which zero pronouns are centered, was only 7.6%. This is because our method preferred NULL phrases over unreliable phrases appearing before the predicate sentence. In fact, the analyzer output only 488 arguments, although the answer was 3,063. To control the NULL preference is a future work for our analyzer.

4 Discussions and Conclusions

We proposed a predicate-argument structure analysis that simultaneously conducts zero-anaphora resolution. By adding noun phrases as candidate arguments that are not only in the sentence of the target predicate but also outside of the sentence, our analyzer identified arguments regardless of whether they appear in the sentence or not. Because we adopted discriminative models for argument identification, we can easily add new features. By using this property, we added language model scores as well as contextual features. We also used contextual information to restrict candidate arguments. As a result, we achieved predicate-argument accuracy of 59.4%, and accuracies of argument identification were F-scores of 73.2%–89.2%.

Verifying argument structures by language models evokes selectional preference of case frames. Sasano et al. (2008) has proposed statistical models using case frames built from 1.6 B sentences. Because the amount of the resources used in our study is quite different, we cannot directly compare the methods and results. However, because our analyzer has scalability that can freely add new features, for our future work, we hope to adopt the case frames as new features and compare their effect.

References

Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proceedings of the Linguistic Annotation Workshop in ACL-2007*, pages 132–139.

Zheng Ping Jiang and Hwee Tou Ng. 2006. Semantic role labeling of nombank: A maximum entropy approach. In *Proceedings of EMNLP-2006*, pages 138–145.

Shigeko Nariyama. 2002. Grammar for ellipsis resolution in Japanese. In *Proceedings of TMI-2002*, pages 135–145.

Ryohei Sasano, Daisuke Kawahara, and Sadao Kurohashi. 2008. A fully-lexicalized probabilistic model for Japanese zero anaphora resolution. In *Proceedings of COLING-2008*, pages 769–776.

Hirotoshi Taira, Sanae Fujita, and Masaaki Nagata. 2008. A Japanese predicate argument structure analysis using decision lists. In *Proceedings of EMNLP-2008*, pages 523–532.

Predicting Barge-in Utterance Errors by using Implicitly Supervised ASR Accuracy and Barge-in Rate per User

Kazunori Komatani
Graduate School of Informatics
Kyoto University
Yoshida, Sakyo, Kyoto 606-8501, Japan
komatani@i.kyoto-u.ac.jp

Alexander I. Rudnicky
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213, U.S.A.
air@cs.cmu.edu

Abstract

Modeling of individual users is a promising way of improving the performance of spoken dialogue systems deployed for the general public and utilized repeatedly. We define "implicitly-supervised" ASR accuracy per user on the basis of responses following the system's explicit confirmations. We combine the estimated ASR accuracy with the user's barge-in rate, which represents how well the user is accustomed to using the system, to predict interpretation errors in barge-in utterances. Experimental results showed that the estimated ASR accuracy improved prediction performance. Since this ASR accuracy and the barge-in rate are obtainable at runtime, they improve prediction performance without the need for manual labeling.

1 Introduction

The automatic speech recognition (ASR) result is the most important input information for spoken dialogue systems, and therefore, its errors are critical problems. Many researchers have tackled this problem by developing ASR confidence measures based on utterance-level information and dialogue-level information (Litman et al., 1999; Walker et al., 2000). Especially in systems deployed for the general public such as those of (Komatani et al., 2005) and (Raux et al., 2006), the systems need to correctly detect interpretation errors caused by various utterances made by various kinds of users including novices. Furthermore, since some users access such systems repeatedly (Komatani et al., 2007), error detection by using individual user models would be a promising way of improving performance.

In another aspect in dialogue systems, certain dialogue patterns indicate that ASR results in certain positions are reliable. For example, Sudoh and Nakano (2005) proposed "post-dialogue confidence scoring" in which ASR results corresponding to the user's intention upon dialogue completion are assumed to be correct and are used for confidence scoring. Bohus and Rudnicky (2007) proposed "implicitly-supervised learning" in which users' responses following the system's explicit confirmations are used for confidence scoring. If ASR results can be regarded as reliable after the dialogue, machine learning algorithms can use such ASR results as teacher signals. This approach enables the system to improve its performance without any manual labeling or transcription, a task which requires much time and labor when spoken dialogue systems are developed.

We focus on users' affirmative and negative responses to the system's explicit confirmations as in (Bohus and Rudnicky, 2007) and estimate the user's ASR accuracy on the basis of his or her history of responses. The estimated ASR accuracy is combined with the user's barge-in rate to predict the interpretation error in the current barge-in utterance. Because the estimated ASR accuracy and the barge-in rate per user are obtainable at runtime, it is possible to improve prediction performance without any manual transcription or labeling.

2 Implicitly Supervised Estimation of ASR Accuracy

2.1 Predicting Errors in Barge-in Utterance

We aim to predict interpretation errors in barge-in utterances at runtime. These errors are caused by ASR errors, and barge-in utterances are more prone to be misrecognized. A user study conducted by Rose and Kim (2003) revealed that there are many more disfluencies when users barge-in compared with when users wait until the system prompt ends. It is difficult to select the erroneous utterances to be rejected by using a classifier that

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 89–92,
Suntec, Singapore, 4 August 2009.

distinguishes speech from noise on the basis of the Gaussian Mixture Model (Lee et al., 2004); such disfluencies and resulting utterance fragments are parts of human speech.

Barge-in utterances are, therefore, more difficult to recognize correctly, especially when novice users barge-in. To detect their interpretation errors, other features should be incorporated instead of speech signals or ASR results. We predicted the interpretation errors in barge-in utterances on the basis of each user's barge-in rate (Komatani et al., 2008). This rate intuitively corresponds to how well users are accustomed to using the system, especially to its barge-in function.

Furthermore, we utilize a user's ASR accuracy in his or her history of all utterances including barge-ins. The ASR accuracy also indicates the user's habituation. However, it has been shown that the user's ASR accuracy and barge-in rate do not improve simultaneously (Komatani et al., 2007). In fact, some expert users have low barge-in rates. We thus can predict whether a barge-in utterance will be correctly interpreted or not by integrating the user's current ASR accuracy and barge-in rate.

2.2 Estimating ASR Accuracy by using Implicitly Supervised Labels

To perform runtime prediction, we use information derived from the dialogue patterns to estimate the user's ASR accuracy. We estimate the accuracy on the basis of the user's history of responses following the system's explicit confirmations such as "Leaving from Kyoto Station. Is that correct?"

Specifically, we assume that the ASR results of affirmative or negative responses following explicit confirmations are correct and that the user utterances corresponding to the content of the affirmative responses are also correct. We further assume that the remaining utterances are incorrect because users do not often respond with "no" for explicit confirmations containing incorrect content and instead repeat their original utterances. Consequently, we regard that the ASR results of the following utterances are correct: (1) affirmative responses and their immediately preceding utterances and (2) negative responses. Accordingly, all other utterances are incorrect. We thus calculate the user's estimated ASR accuracy by using the user's utterance history, as follows:

$$\begin{aligned}&(\text{Estimated ASR accuracy})\\ &= \frac{2 \times (\#\text{affirmatives}) + (\#\text{negatives})}{(\#\text{all utterances})} \qquad (1)\end{aligned}$$

2.3 Predicting Errors by Using Barge-in Rate and ASR Accuracy

We predict the errors in barge-in utterances by using a logistic regression function:

$$P = \frac{1}{1 + \exp(-(a_1 x_1 + a_2 x_2 + b))}.$$

Its inputs x_1 and x_2 are the barge-in rate until the current utterance and ASR accuracy until the previous utterance. To account for temporal changes in barge-in rates, we set a window when calculating them (Komatani et al., 2008). That is, when the window width is N, the rates are calculated by using only the last N utterances, and the previous utterances are discarded. When the window width exceeds the total number of utterances by the user, the barge-in rates are calculated by using all the user's utterances. Thus, when the width exceeds 2,838, the maximum number of utterances made by one user in our data, the barge-in rates equal the average rates of all previous utterances by the user.

We calculate the estimated ASR accuracy every time a user makes an affirmative or negative response. When the user makes other utterances, we take the estimated accuracy when the *last* affirmative/negative response is made to be the accuracy of those utterances.

3 Experimental Evaluation

3.1 Target Data

We used data collected by the Kyoto City Bus Information System (Komatani et al., 2005). This system locates a bus that a user wants to ride and tells the user how long it will be before the bus arrives. The system was accessible to the public by telephone. It used the safest strategy to prevent erroneous responses, that is, to make explicit confirmations for all ASR results.

We used 27,519 utterances after removing calls whose phone numbers were not recorded and those the system developer called for debugging. From that number, there were 7,193 barge-in utterances, i.e., utterances that a user starts speaking during a system prompt. The phone numbers of the calls were recorded, and we assumed that each

Table 1: ASR accuracy by response type

	Correct	Incorrect	Total	(Acc.)
Affirmative	9,055	246	9,301	(97.4%)
Negative	2,006	289	2,295	(87.4%)
Other	8,914	7,009	15,923	(57.9%)
Total	19,975	7,544	27,519	(72.6%)

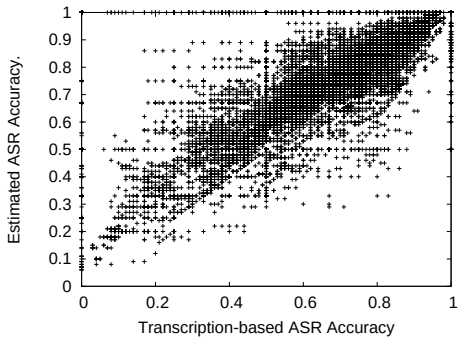


Figure 1: Correlation between transcription-based and estimated ASR accuracy

number corresponded to one individual. Most of the numbers were those of mobile phones, which are usually not shared, so the assumption seems reasonable.

Each utterance was transcribed and its interpretation result, correct or not, was given manually. We assumed that an interpretation result for an utterance was correct if all content words in its transcription were correctly included in the result. The result was regarded as an error if any content words were missed or misrecognized.

3.2 Verifying Implicitly Supervised Labels

We confirmed our assumption that the ASR results of affirmative or negative responses following explicit confirmations are correct. We classified the user utterances into affirmatives, negatives, and other, and calculated the ASR accuracies (precision rates) as shown in Table 1. Affirmatives include *hai* ('yes'), *soudesu* ('that's right'), OK, etc; and negatives include *iie* ('no'), *chigaimasu* ('I don't agree'), *dame* ('No good'), etc. The table indicates that the ASR accuracies of affirmatives and negatives were high. One of the reasons for the high accuracy was that these utterances are much shorter than other content words, so they were not confused with other content words. Another reason was that the system often gave help messages such as "Please answer *yes* or *no*."

We then analyzed the correlation between the transcription-based ASR accuracy and the estimated ASR accuracy based on Equation 1. We plotted the two ASR accuracies in Figure 1 for 26,231 utterances made after at least one affirmative/negative response by the user. The correlation coefficient between them was 0.806. Although the assumption that all ASR results of affirmative/negative responses are correct might be strong, the estimated ASR accuracy had a high correlation with the transcription-based ASR accuracy.

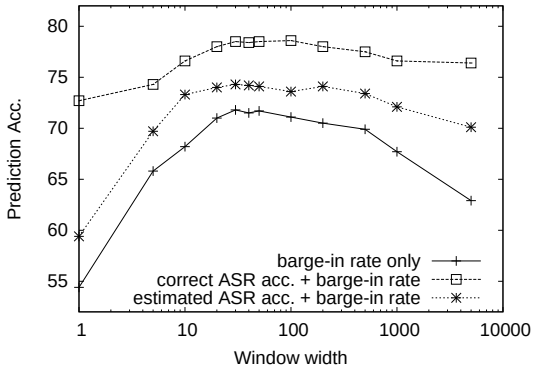


Figure 2: Prediction accuracy with various window widths

3.3 Prediction using Implicitly Supervised Labels

We measured the prediction accuracy for 7,193 barge-in utterances under several conditions. We did not set windows when calculating the ASR accuracies and thus used all previous utterances of the user, because the windows did not improve prediction accuracy. One of the reasons for this lack of improvement is that the ASR accuracies did not change as significantly as the barge-in rates because the accuracies of frequent users converged earlier (Komatani et al., 2007).

We first confirmed the effect of the transcription-based ("correct", hereafter) ASR accuracy. As shown in Figure 2 and Table 2, the prediction accuracy improved by using the ASR accuracy in addition to the barge-in rate. The best prediction accuracy (78.6%) was when the window width of the barge-in rate was 100, and the accuracy converged when the width was 30. The prediction accuracy was 72.7% when only the "correct" ASR accuracy was used, and the prediction accuracy was 71.8% when only the barge-in rate was used. Thus, the prediction accuracy was better when both inputs were used rather than when either input was used. This

Table 2: Best prediction accuracies for each condition and window width w

Conditions (Used inputs)	Prediction acc. (%)
barge-in rate	71.8 (w=30)
correct ASR acc.	72.7
+ barge-in rate	78.6 (w=100)
estimated ASR acc.	59.4
+ barge-in rate	74.3 (w=30)

fact indicates that both the barge-in rate and ASR accuracy have different information and contribute to the prediction accuracy.

Next, we analyzed the prediction accuracy after replacing the correct ASR accuracy with the estimated one described in Section 2.2. The best accuracy (74.3%) was when the window width was 30. This accuracy was higher than that of using only barge-in rates. Hence, the estimated ASR accuracy without manual labeling is effective in predicting the errors in barge-in utterances at runtime.

4 Conclusion

We proposed a method to estimate the errors in barge-in utterances by using a novel dialogue-level feature obtainable at runtime. This method does not require supervised manual labeling. The estimated ASR accuracy based on the user's utterance history was dependable in predicting the errors in the current utterance. We thus showed that ASR accuracy can be estimated in an implicitly supervised manner.

The information obtained by our method can be used for confidence scoring. Thus, our future work will include integrating the proposed features with bottom-up information such as acoustic-score-based confidence measures. Additionally, we simply assumed in this study that all affirmative and negative responses following the explicit confirmation are correct. By modeling this assumption more precisely, prediction accuracy will improve. Finally, we identified individuals on the basis of their telephone numbers. If we utilize user identification techniques to account for situations when no speaker information is available beforehand, this method can be applied to systems other than telephone-based ones, e.g., to human-robot interaction.

Acknowledgments

We are grateful to Prof. Tatsuya Kawahara of Kyoto University who led the project of the Kyoto City Bus Information System.

References

Dan Bohus and Alexander Rudnicky. 2007. Implicitly-supervised learning in spoken language interfaces: an application to the confidence annotation problem. In *Proc. SIGdial Workshop on Discourse and Dialogue*, pages 256–264.

Kazunori Komatani, Shinichi Ueno, Tatsuya Kawahara, and Hiroshi G. Okuno. 2005. User modeling in spoken dialogue systems to generate flexible guidance. *User Modeling and User-Adapted Interaction*, 15(1):169–183.

Kazunori Komatani, Tatsuya Kawahara, and Hiroshi G. Okuno. 2007. Analyzing temporal transition of real user's behaviors in a spoken dialogue system. In *Proc. INTERSPEECH*, pages 142–145.

Kazunori Komatani, Tatsuya Kawahara, and Hiroshi G. Okuno. 2008. Predicting ASR errors by exploiting barge-in rate of individual users for spoken dialogue systems. In *Proc. INTERSPEECH*, pages 183–186.

Akinobu Lee, Keisuke Nakamura, Ryuichi Nisimura, Hiroshi Saruwatari, and Kiyohiro Shikano. 2004. Noice robust real world spoken dialogue system using GMM based rejection of unintended inputs. In *Proc. Int'l Conf. Spoken Language Processing (ICSLP)*, pages 173–176.

Diane J. Litman, Marilyn A. Walker, and Michael S. Kearns. 1999. Automatic detection of poor speech recognition at the dialogue level. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 309–316.

Antoine Raux, Dan Bohus, Brian Langner, Alan W. Black, and Maxine Eskenazi. 2006. Doing research on a deployed spoken dialogue system: One year of Let's Go! experience. In *Proc. INTERSPEECH*.

Richard C. Rose and Hong Kook Kim. 2003. A hybrid barge-in procedure for more reliable turn-taking in human-machine dialog systems. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 198–203.

Katsuhito Sudoh and Mikio Nanano. 2005. Post-dialogue confidence scoring for unsupervised statistical language model training. *Speech Communication*, 45:387–400.

Marilyn Walker, Irene Langkilde, Jerry Wright, Allen Gorin, and Diane Litman. 2000. Learning to predict problematic situations in a spoken dialogue system: Experiments with How May I Help You? In *Proc. North American Chapter of Association for Computational Linguistics (NAACL)*, pages 210–217.

Automatic Generation of Information-seeking Questions Using Concept Clusters

Shuguang Li
Department of Computer Science
University of York, YO10 5DD, UK
sgli@cs.york.ac.uk

Suresh Manandhar
Department of Computer Science
University of York, YO10 5DD, UK
suresh@cs.york.ac.uk

Abstract

One of the basic problems of efficiently generating information-seeking dialogue in interactive question answering is to find the topic of an information-seeking question with respect to the answer documents. In this paper we propose an approach to solving this problem using concept clusters. Our empirical results on TREC collections and our ambiguous question collection shows that this approach can be successfully employed to handle ambiguous and list questions.

1 Introduction

Question Answering systems have received a lot of interest from NLP researchers during the past years. But it is often the case that traditional QA systems cannot satisfy the information needs of the users as the question processing part may fail to properly classify the question or the information needed for extracting and generating the answer is either implicit or not present in the question. In such cases, interactive dialogue is needed to clarify the information needs and reformulate the question in a way that will help the system to find the correct answer.

Due to the fact that casual users often ask questions with ambiguity and vagueness, and most of the questions have multiple answers, current QA systems return a list of answers for most questions. The answers for one question usually belong to different topics. In order to satisfy the information needs of the user, information-seeking dialogue should take advantage of the inherent grouping of the answers.

Several methods have been investigated for generating topics for questions in information-seeking dialogue. Hori et al. (2003) proposed a method for generating the topics for disambiguation questions. The scores are computed purely based on the syntactic ambiguity present in the question. Phrases that are not modified by other phrases are considered to be highly ambiguous while phrases that are modified are considered less ambiguous. Small et al. (2004) utilizes clarification dialogue to reduce the misunderstanding of the questions between the HITIQA system and the user. The topics for such clarification questions are based on manually constructed topic frames. Similarly in (Hickl et al., 2006), suggestions are made to users in the form of predictive question and answer pairs (known as QUABs) which are either generated automatically from the set of documents returned for a query (using techniques first described in (Harabagiu et al., 2005), or are selected from a large database of questions-answer pairs created offline (prior to a dialogue) by human annotators. In Curtis et al. (2005), query expansion of the question based on Cyc Knowledge is used to generate topics for clarification questions. In Duan et al. (2008), the tree-cutting model is used to select topics from a set of relevant questions from Yahoo Answers.

None of the above methods consider the contexts of the list of answers in the documents returned by QA systems. The topic of a good information-seeking question should not only be relevant to the original question but also should be able to distinguish each answer from the others so that the new information can reduce the ambiguity and vagueness in the original question. Instead of using traditional clustering methods on categorization of web results, we present a new topic generation approach using concept clusters and a separability scoring mechanism for ranking the topics.

2 Topic Generation Based on Concept Clustering

Text categorization and clustering especially hierarchical clustering are predominant approaches to organizing large amounts of information into top-

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 93–96,
Suntec, Singapore, 4 August 2009.

ics or categories. But the main issue of categorization is that it is still difficult to automatically construct a good category structure, and manually formed hierarchies are usually small. And the main challenge of clustering algorithms is that the automatically formed cluster hierarchy may be unreadable or meaningless for human users. In order to overcome the limits of the above methods, we propose a concept clusters method and choose the labels of the clusters as topics.

Recent research on automatically extracting concepts and clusters of words from large database makes it feasible to grow a big set of concept clusters. Clustering by Committee (CBC) in Pantel et al. (2002) made use of the fact that words in the same cluster tend to appear in similar contexts. Pasca et al. (2008) utilized Google logs and lexico-syntactic patterns to get clusters with labels simultaneously. Google also released Google Sets which can be used to grow concept clusters with different sizes.

Currently our clusters are the union of the sets generated by the above three approaches, and we label them using the method described in Pasca et al. (2008). We define the **concept clusters** in our collection as $\{C_1, C_2, ..., C_n\}$. C_i=$\{e_{i1}, e_{i2}, ..., e_{im}\}$, e_{ij} is j^{th} subtopic of cluster C_i and m is the size of C_i.

We designed our system to take a question and its corresponding list of answers as input and then retrieve Google snippet documents for each of the answers with respect to the question. In a vectorspace model, a document is represented by a vector of keywords extracted from the document, with associated weights representing the importance of the keywords in the document and within the whole document collection. A document D_j in the collection is represented as $\{W_{0j}, W_{1j}, ..., W_{nj}\}$, and W_{ij} is the weight of word i in document j. Here we use our concept clusters to create concept cluster vectors. A document D_j now is represented as $<WC_{1j}, WC_{2j}, ..., WC_{nj}>$, and WC_{ij} is the score vector of document D_j for concept cluster C_i:

WC_{ij} = $<Score_j(e_{i1}), Score_j(e_{i2}), ...Score_j(e_{im})>$

$Score_j(e_{ip})$ is the weight of subtopic e_{ip} of cluster C_i in document D_j.

Currently we use tf-idf scheme (Yang et al., 1999) to calculate the weight of subtopics.

3 Concept Cluster Separability Measure

We view different concept clusters from the contexts of the answers as different groups of features that can be used to classify the answers documents. We rank different context features by their separability on the answers. Currently our system retrieves the answers from Google search snippets, and each snippet is quite short. So we combine the top 50 snippets for one answer into one document. One answer is associated with one such big document. We propose the following interclass measure to compare the separability of different clusters:

$$Score(C_i) = \frac{D}{N} \sum_{p<q}^{N} Dis(D_p, D_q),$$

D is the Dimension Penalty score, $D = \frac{1}{M}$,
M is the size of cluster C_i,
N is the combined total number of classes from all the answers

$$Dis(D_p, D_q) = \sqrt{\sum_{m=0}^{n} (Score_p(e_{im}) - Score_q(e_{im}))^2}$$

We introduce D, the "Dimension Penalty" score which gives higher penalty to bigger clusters. Currently we use the reciprocal of the size of the cluster. The second part is the average pairwise distance between answers. N is the total number of classes of the answers. Next we describe in detail how to use the concept cluster vectors and separability measure to rank clusters.

4 Cluster Ranking Algorithm

Input:
Answer set $A = \{A_1, A_2, ..., A_p\}$;
Documents set $D = \{D_1, D_2, ..., D_p\}$ associated with answer set A;
Concept cluster set $CS = \{C_i \mid$ some of the subtopics from C_i occurs in $D\}$;
Threshold Θ_1, Θ_2; The question Q;
Concept cluster set $QS = \{C_i \mid$ some of the subtopics from C_i occurs in $Q\}$
Output:
$T = \{< C_i, Score >\}$, a set of pairs of a concept cluster and its ranking score;
QS;
Variables: X, Y;
Steps:

1. $CS = CS - QS$
2. For each cluster C_i in CS
3. X = No. of answers in which context subtopics from C_i are present;
4. Y = No. of subtopics from C_i that occurs in the answers' contexts;
5. If $X < \Theta_1$ or $Y < \Theta_2$
6. delete C_i from CS
7. continue
8. Represent every document as a concept cluster vector on C_i (see section 2)
9. Calculate the Score(C_i) using our separability measure
10. Store $< C_i, Score >$ in T
11. return T the medoid.

Figure 1: Concept Cluster Ranking Algorithm

Figure 1 describes the algorithm for ranking concept clusters based on their separability score. This algorithm starts by deleting all

the clusters which are in QS from CS so that we only focus on the context clusters whose subtopics are present in the answers. However in some cases this assumption is incorrect[1]. Taking the question shown in Table 2 for example, there are 6 answers for question LQ1, and in **Step 1** CS = {C_{41}American State, C_{1522}Times, C_{414}Tournament, $C_{10004}Year$, ...} and QS = {C_{4545}Event}. Using cluster C_{414} (see Table 2), D = {D_1{Daytona 500, 24 Hours of Daytona, 24 Hours of Le Mans, ...}, D_2{3M Performance 400, Cummins 200, ...}, D_3{Indy 500, Truck series, ...}, ...}, and hence the vector representation for a given document D_j using C_{414} will be $<Score_j(indy\,500), Score_j(Cummins\,200), Score_j(daytona\,500), ...>$.

In **Step 2** through **11** from Figure 1, for each context cluster C_i in CS we calculate X (the number of answers in which context subtopics from C_i are present), and Y (the number of subtopics from C_i that occurs in the answers' contexts). We would like the clusters to hold two characteristics: (a) at least occur in Θ_1 answers as we want to have a cluster whose subtopics are widely distributed in the answers. Currently we set Θ_1 as half the number of the answers; (b) at least have Θ_2 subtopics occurring in the answers' documents. We set Θ_2 as the number of the answers. For example, for cluster C_{414}, X = 6, Y = 10, Θ_1 = 3 and Θ_2 = 6, so this cluster has the above two characteristics. If a cluster has the above two characteristics, we use our separability measure described in section 3 to calculate a score for this cluster. The size of C_{414} is 11, so $Score(C_{414}) = \frac{1}{11\times 6}\sum_{p<q}^{N} Dis(D_p, D_q)$. Ranking the clusters based on this separability score means we will select a cluster which has several subtopics occurring in the answers and the answers are distinguished from each other because they belong to these different subtopics. The top three clusters for question LQ1 is shown in Table 2.

5 Experiment

5.1 Data Set and Baseline Method

To the best of our knowledge, the only available test data of multiple answer questions are list questions from TREC 2004-2007 Data. For our first list question collection we randomly selected 200 questions which have at least 3 answers. We changed the list questions to factoid ones with additional words from their context questions to eliminate ellipsis and reference. For the ambiguous questions, we manually choose 200 questions from TREC 1999-2007 data and some questions discussed as examples in Hori et al. (2003) and Burger et al. (2001).

We compare our approach with a baseline method. Our baseline system does not rank the clusters by the above separability score instead it prefers the cluster which occurs in more answers and have more subtopics distributed in the answer documents. If we still use X to represent the number of answers in which context subtopics from one cluster are present and Y to represent the number of subtopics from this cluster that occurs in the answers' contexts, for the baseline system, we will use $X \times Y$ to rank all the concept clusters found in the contexts.

5.2 Results and Error Analysis

We applied our algorithm on the two collections of questions. Two assessors were involved in the manual judgments with an inter-rater agreement of 97%. For each approach, we obtained the top 20 clusters based on their scores. Given a cluster with its subtopics in the contexts of the answers, an assessor manually labeled each cluster 'good' or 'bad'. If it is labeled 'good', the cluster is deemed relevant to the question and the cluster's label could be used as dialogue seeking question's topic to distinguish one answer from the others. Otherwise, the assessor will label a cluster as 'bad'. We use the above two ranking approaches to rank the clusters for each question. Table 1 provides the statistics of the performance on the the two question collection. List_B means the baseline method on the list question set while Ambiguous_S means our separability method on the ambiguous questions. The 'MAP' column is the mean of average precisions over the set of clusters. The 'P@1' column is the precision of the top one cluster while the 'P@3' column is the precision of the top three clusters[2]. The 'Err@3' column is the percentage of questions whose top three clusters are all labeled 'bad'. One example associated with the manually constructed desirable questions

[1]For the question "In which movies did Christopher Reeve acted?", cluster Actor{Christopher Reeve, michael caine, anthony hopkins, ...} is quite useful. While for "Which country won the football world cup?" cluster Sports{football, hockey, ...} is useless.

[2]'P@3' is the number of 'good' clusters out of the top three clusters

Table 1: Experiment results

Methods	MAP	P@1	P@3	Err@3
List_B	41.3%	42.1%	27.7%	33.0%
List_S	60.3%	90.0%	81.3%	11.0%
Ambiguous_B	31.1%	33.2%	21.8%	47.1%
Ambiguous_S	53.6%	71.1%	64.2%	29.7%

Table 2: TREC Question Examples

LQ1:	Who is the winners of the NASCAR races?
1^{st}	C_{414}(Tournament):{indy 500, Cummins 200, daytona 500, ...}
Q1	Which Tournament are you interested in?
2^{nd}	C_{41}(American State):{houston, baltimore, los angeles, ...}
Q2	Which American State were the races held?
3^{rd}	C_{1522}(Times):{once, twice, three times, ...}
Q3	How many times did the winner win?

is shown in Table 2.

From Table 1, we can see that our approach outperforms the baseline approach in terms of all the measures. We can see that 11% of the questions have no 'good' clusters. Further analysis of the answer documents shows that the 'bad' clusters fall into four categories. First, there are noisy subtopics in some clusters. Second, some questions' clusters are all labeled 'bad' because the contexts for different answers are too similar. Third, unstructured web document soften contain multiple subtopics. This means that different subtopics are in the context of the same answer. Currently we only look for context words while not using any scheme to specify whether there is a relationship between the answer and the subtopics. Finally, for other 'bad' cases and the questions with no good clusters all of the separability scores are quite low. This is because the answers fall into different topics which do not share a common topic in our cluster collection.

6 Conclusion and Discussion

This paper proposes a new approach to solve the problem of generating an information-seeking question's topic using concept clusters that can be used in a clarification dialogue to handle ambiguous questions. Our empirical results show that this approach leads to good performance on TREC collections and our ambiguous question collections. The contribution of this paper are: (1) a new concept cluster method that maps a document into a vector of subtopics; (2) a new ranking scheme to rank the context clusters according to their separability. The labels of the chosen clusters can be used as topics in an information-seeking question. Finally our approach shows significant improvement (nearly 48% points) over comparable baseline system.

But currently we only consider the context clusters while ignoring the clusters associated with the questions. In the future, we will further investigate the relationships between the concept clusters in the question and the answers.

References

Tiphaine Dalmas, Bonnie L. Webber: Answer comparison in automated question answering. J. Applied Logic (JAPLL) 5(1):104-120, (2007).

Chiori Hori, Sadaoki Furui: A new approach to automatic speech summarization. IEEE Transactions on Multimedia (TMM) 5(3):368-378, (2003).

Sharon Small and Tomek Strzalkowski, HITIQA: A Data Driven Approach to Interactive Analytical Question Answering, in Proceedings of HLT-NAACL 2004: Short Papers, (2004).

Andrew Hickl, Patrick Wang, John Lehmann, Sanda M. Harabagiu: FERRET: Interactive Question-Answering for Real-World Environments. ACL, (2006).

Sanda M. Harabagiu, Andrew Hickl, John Lehmann, Dan I. Moldovan: Experiments with Interactive Question-Answering. ACL, (2005).

John Burger et al.: Issues, Tasks and Program Structures to Roadmap Research in Question and Answering (Q&A),DARPA/NSF committee publication, (2001).

Patrick Pantel, Dekang Lin: Document clustering with committees. SIGIR 2002:199-206, (2002).

Marius Pasca and Benjamin Van Durme: Weakly-Supervised Acquisition of Open-Domain Classes and Class Attributes from Web Documents and Query Logs. ACL, (2008).

Sanda M. Harabagiu, Andrew Hickl, V. Finley Lacatusu: Satisfying information needs with multi-document summaries. Inf. Process. Manage. (IPM) 43(6):1619-1642, (2007).

Huizhong Duan, Yunbo Cao, Chin-Yew Lin and Yong Yu: Searching Questions by Identifying Question Topic and Question Focus. ACL, (2008).

Jon Curtis, G. Matthews and D. Baxter: On the Effective Use of Cyc in a Question Answering System. IJCAI Workshop on Knowledge and Reasoning for Answering Questions, Edinburgh, (2005).

Correlating Human and Automatic Evaluation of a German Surface Realiser

Aoife Cahill
Institut für Maschinelle Sprachverarbeitung (IMS)
University of Stuttgart
70174 Stuttgart, Germany
aoife.cahill@ims.uni-stuttgart.de

Abstract

We examine correlations between native speaker judgements on automatically generated German text against automatic evaluation metrics. We look at a number of metrics from the MT and Summarisation communities and find that for a relative ranking task, most automatic metrics perform equally well and have fairly strong correlations to the human judgements. In contrast, on a naturalness judgement task, the General Text Matcher (GTM) tool correlates best overall, although in general, correlation between the human judgements and the automatic metrics was quite weak.

1 Introduction

During the development of a surface realisation system, it is important to be able to quickly and automatically evaluate its performance. The evaluation of a string realisation system usually involves string comparisons between the output of the system and some gold standard set of strings. Typically automatic metrics from the fields of Machine Translation (e.g. BLEU) or Summarisation (e.g. ROUGE) are used, but it is not clear how successful or even appropriate these are. Belz and Reiter (2006) and Reiter and Belz (2009) describe comparison experiments between the automatic evaluation of system output and human (expert and non-expert) evaluation of the same data (English weather forecasts). Their findings show that the NIST metric correlates best with the human judgements, and all automatic metrics favour systems that generate based on frequency. They conclude that automatic evaluations should be accompanied by human evaluations where possible. Stent et al. (2005) investigate a number of automatic evaluation methods for generation in terms of adequacy and fluency on automatically generated English paraphrases. They find that the automatic metrics are reasonably good at measuring adequacy, but not good measures of fluency, i.e. syntactic correctness.

In this paper, we carry out experiments to correlate automatic evaluation of the output of a surface realisation ranking system for German against human judgements. We particularly look at correlations at the individual sentence level.

2 Human Evaluation Experiments

The data used in our experiments is the output of the Cahill et al. (2007) German realisation ranking system. That system is couched within the Lexical Functional Grammar (LFG) grammatical framework. LFG has two levels of representation, C(onstituent)-Structure which is a context-free tree representation and F(unctional)-Structure which is a recursive attribute-value matrix capturing basic predicate-argument-adjunct relations.

Cahill et al. (2007) use a large-scale handcrafted grammar (Rohrer and Forst, 2006) to generate a number of (almost always) grammatical sentences given an input F-Structure. They show that a linguistically-inspired log-linear ranking model outperforms a simple baseline tri-gram language model trained on the Huge German Corpus (HGC), a corpus of 200 million words of newspaper and other text.

Cahill and Forst (2009) describe a number of experiments where they collect judgements from native speakers about the three systems compared in Cahill et al. (2007): (i) the original corpus string, (ii) the string chosen by the language model, and (iii) the string chosen by the linguistically-inspired log-linear model.[1] We only take the data from 2 of those experiments since the remaining experiments would not provide any

[1] In all cases, the three strings were different.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 97–100,
Suntec, Singapore, 4 August 2009.

informative correlations. In the first experiment that we consider (A), subjects are asked to rank on a scale from 1–3 (1 being the best, 3 being the worst) the output of the three systems (joint rankings were not permitted). In the second experiment (B), subjects were asked to rank on a scale from 1–5 (1 being the worst, 5 being the best) how natural sounding the string chosen by the log-linear model was. The goal of experiment B was to determine whether the log-linear model was choosing good or bad alternatives to the original string. Judgements on the data were collected from 24 native German speakers. There were 44 items in Experiment A with an average sentence length of 14.4, and there were 52 items in Experiment B with an average sentence length of 12.1. Each item was judged by each native speaker at least once.

3 Correlation with Automatic Metrics

We examine the correlation between the human judgements and a number of automatic metrics:

BLEU (Papineni et al., 2001) calculates the number of *n*-grams a solution shares with a reference, adjusted by a brevity penalty. Usually the geometric mean for scores up to 4-gram are reported.

ROUGE (Lin, 2004) is an evaluation metric designed to evaluate automatically generated summaries. It comprises a number of string comparison methods including *n*-gram matching and skip-*n*grams. We use the default ROUGE-L longest common subsequence f-score measure.[2]

GTM General Text Matching (Melamed et al., 2003) calculates word overlap between a reference and a solution, without double counting duplicate words. It places less importance on word order than BLEU.

SED Levenshtein (String Edit) distance

WER Word Error Rate

TER Translation Error Rate (Snover et al., 2006) computes the number of insertions, deletions, substitutions and shifts needed to match a solution to a reference.

Most of these metrics come from the Machine Translation field, where the task is arguably significantly different. In the evaluation of a surface realisation system (as opposed to a complete generation system), typically the choice of vocabulary is limited and often the task is closer to word reordering. Many of the MT metrics have methods for attempting to account for different but equivalent translations of a given source word, typically by integrating a lexical resource such as WordNet. Also, these metrics were mostly designed to evaluate English output, so it is not clear that they will be equally appropriate for other languages, especially freer word order ones, such as German.

[2]Preliminary experiments with the skip *n*-grams performed worse than the default parameters.

	Experiment A			Experiment B
	GOLD	LM	LL	LL
human A (rank 1–3)	1.4	2.55	2.05	
human B (scale 1–5)				3.92
BLEU	1.0	0.67	0.72	0.79
ROUGE-L	1.0	0.85	0.78	0.85
GTM	1.0	0.55	0.60	0.74
SED	1.0	0.54	0.61	0.71
WER	0.0	48.04	39.88	28.83
TER	0.0	0.16	0.14	0.11
DEP	100	82.60	87.50	93.11
WDEP	1.0	0.70	0.82	0.90

Table 1: Average scores of each metric for Experiment A data

	Sentence		Corpus	
	corr	p-value	corr	p-value
BLEU	-0.615	<0.001	-1	0.3333
ROUGE-L	-0.644	<0.001	-0.5	1
GTM	-0.643	<0.001	-1	0.3333
SED	-0.628	<0.001	-1	0.3333
WER	0.623	<0.001	1	0.3333
TER	0.608	<0.001	1	0.3333

Table 2: Correlation between human judgements for experiment A (rank 1–3) and automatic metrics

The scores given by each metric for the data used in both experiments are presented in Table 1. For the Experiment A data, we use the Spearman rank correlation coefficient to measure the correlation between the human judgements and the automatic scorers. The results are presented in Table 2 for both the sentence and the corpus level correlations, we also present p-values for statistical significance. Since we only have judgements on three systems, the corpus correlation is not that informative. Interestingly, the ROUGE-L metric is the only one that does not rank the output of the three systems in the same order as the judges. It ranks the strings chosen by the language model higher than the strings chosen by the log-linear model. However, at the level of the individual sentence, the ROUGE-L metric correlates best with the human judgements. The GTM metric correlates at about the same level, but in general there seems to be little difference between the metrics.

For the Experiment B data we use the Pearson correlation coefficient to measure the correlation between the human judgements and the automatic

	Sentence Correlation	P-Value
BLEU	0.095	0.5048
ROUGE-L	0.207	0.1417
GTM	0.424	0.0017
SED	0.168	0.2344
WER	-0.188	0.1817
TER	-0.024	0.8646

Table 3: Correlation between human judgements for experiment B (naturalness scale 1–5) and automatic metrics

metrics. The results are given in Table 3. Here we only look at the correlation at the individual sentence level, since we are looking at data from only one system. For this data, the GTM metric clearly correlates most closely with the human judgements, and it is the only metric that has a statistically significant correlation. BLEU and TER correlate particularly poorly, with correlation coefficients very close to zero.

3.1 Syntactic Metrics

Recently, there has been a move towards more syntactic, rather than purely string based, evaluation of MT output and summarisation (Hovy et al., 2005; Owczarzak et al., 2008). The idea is to go beyond simple string comparisons and evaluate at a deeper linguistic level. Since most of the work in this direction has only been carried out for English so far, we apply the idea rather than a specific tool to the data. We parse the data from both experiments with a German dependency parser (Hall and Nivre, 2008) trained on the TIGER Treebank (with sentences 8000-10000 heldout for testing). This parser achieves 91.23% labelled accuracy on the 2000-sentence test set.

To calculate the correlation between the human judgements and the dependency parser, we parse the original strings as well as the strings chosen by the log-linear and language models. The standard evaluation procedure relies on both strings being identical to calculate (un-)labelled dependency accuracy, and so we map the dependencies produced by the parser into sets of triples as used in the evaluation software of Crouch et al. (2002) where each dependency is represented as `deprel(head,word)` and each word is indexed with its position in the original string.[3] We compare the parses for both experiments against the parses of the original strings. We calculate both a weighted and unweighted dependency f-score, as given in Table 1. The unweighted f-score is calculated by taking the average of the scores for each dependency type, while the weighted f-score weighs each average score by its frequency in the test corpus. We calculate the Spearman and Pearson correlation coefficients as before; the results are given in Table 4. The results show that the unweighted dependencies correlate more closely (and statistically significantly) with the human judgements than the weighted ones. This suggests that the frequency of a dependency type does not matter as much as its overall correctness.

[3]This is a 1-1 mapping, and the indexing ensures that duplicate words in a sentence are not confused.

	Experiment A		Experiment B	
	corr	p-value	corr	p-value
Dependencies	-0.640	<0.001	0.186	0.1860
Unweighted Deps	-0.657	<0.001	0.290	0.03686

Table 4: Correlation between dependency-based evaluation and human judgements

4 Discussion

The large discrepancy between the absolute correlation coefficients for Experiment A and B can be explained by the fact that they are different tasks. Experiment A ranks 3 strings relative to one another, while Experiment B measures the naturalness of the string. We would expect automatic metrics to be better at the first task than the second, as it is easier to rank systems relative to each other than to give a system an absolute score.

Disappointingly, the correlation between the dependency parsing metric and the human judgements was no higher than the simple GTM string-based metric (although it did outperform all other automatic metrics). This does not correspond to related work on English Summarisation evaluation (Owczarzak, 2009) which shows that a metric based on an automatically induced LFG parser for English achieves comparable or higher correlation with human judgements than ROUGE and Basic Elements (BE).[4] Parsers of German typically do not achieve as high performance as their English counterparts, and further experiments including alternative parsers are needed to see if we can improve performance of this metric.

The data used in our experiments was almost always grammatically correct. Therefore the task

[4]The GTM metric was not compared in that paper

of an evaluation system is to score more natural sounding strings higher than marked or unnatural ones. In this respect, our findings mirror those of Stent et al. (2005) for English data, that the automatic metrics do not correlate well with human judges on syntactic correctness.

5 Conclusions

We presented data that examined the correlation between native speaker judgements and automatic evaluation metrics on automatically generated German text. We found that for our first experiment, all metrics were correlated to roughly the same degree (with ROUGE-L achieving the highest correlation at an individual sentence level and the GTM tool not far behind). At a corpus level all except ROUGE were in agreement with the human judgements. In the second experiment, the General Text Matcher Tool had the strongest correlation. We carried out an experiment to test whether a more sophisticated syntax-based evaluation metric performed better than the more simple string-based ones. We found that while the unweighted dependency evaluation metric correlated with the human judgements more strongly than almost all metrics, it did not outperform the GTM tool. The correlation between the human judgements and the automatic evaluation metrics was much higher for the relative ranking task than for the naturalness task.

Acknowledgments

This work was funded by the Collaborative Research Centre (SFB 732) at the University of Stuttgart. We would like to thank Martin Forst, Alex Fraser and the anonymous reviewers for their helpful feedback. Furthermore, we would like to thank Johan Hall, Joakim Nivre and Yannick Versely for their help in retraining the MALT dependency parser with our data set.

References

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *Proceedings of EACL 2006*, pages 313–320, Trento, Italy.

Aoife Cahill and Martin Forst. 2009. Human Evaluation of a German Surface Realisation Ranker. In *Proceedings of EACL 2009*, pages 112–120, Athens, Greece, March.

Aoife Cahill, Martin Forst, and Christian Rohrer. 2007. Stochastic Realisation Ranking for a Free Word Order Language. In *Proceedings of ENLG-07*, pages 17–24, Saarbrücken, Germany, June.

Richard Crouch, Ron Kaplan, Tracy Holloway King, and Stefan Riezler. 2002. A comparison of evaluation metrics for a broad coverage parser. In *Proceedings of the LREC Workshop: Beyond PARSEVAL*, pages 67–74, Las Palmas, Spain.

Johan Hall and Joakim Nivre. 2008. A dependency-driven parser for German dependency and constituency representations. In *Proceedings of the Workshop on Parsing German*, pages 47–54, Columbus, Ohio, June.

Eduard Hovy, Chin yew Lin, and Liang Zhou. 2005. Evaluating duc 2005 using basic elements. In *Proceedings of DUC-2005*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July.

I. Dan Melamed, Ryan Green, and Joseph P. Turian. 2003. Precision and recall of machine translation. In *Proceedings of NAACL-03*, pages 61–63, NJ, USA.

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2008. Evaluating machine translation with LFG dependencies. *Machine Translation*, 21:95–119.

Karolina Owczarzak. 2009. DEPEVAL(summ): Dependency-based Evaluation for Automatic Summaries. In *Proceedings of ACL-IJCNLP 2009*, Singapore.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL-02*, pages 311–318, NJ, USA.

Ehud Reiter and Anja Belz. 2009. An Investigation into the Validity of Some Metrics for Automatically Evaluating Natural Language Generation Systems. *Computational Linguistics*, 35.

Christian Rohrer and Martin Forst. 2006. Improving Coverage and Parsing Quality of a Large-Scale LFG for German. In *Proceedings of LREC 2006*, Genoa, Italy.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and Ralph Weischedel. 2006. A study of translation error rate with targeted human annotation. In *Proceedings of AMTA 2006*, pages 223–231.

Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating evaluation methods for generation in the presense of variation. In *Proceedings of CICLING*, pages 341–351.

Leveraging Structural Relations for Fluent Compressions at Multiple Compression Rates

Sourish Chaudhuri, Naman K. Gupta, Noah A. Smith, Carolyn P. Rosé
Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA-15213, USA.
{sourishc, nkgupta, nasmith, cprose}@cs.cmu.edu

Abstract

Prior approaches to sentence compression have taken low level syntactic constraints into account in order to maintain grammaticality. We propose and successfully evaluate a more comprehensive, generalizable feature set that takes syntactic and structural relationships into account in order to sustain variable compression rates while making compressed sentences more coherent, grammatical and readable.

1 Introduction

We present an evaluation of the effect of syntactic and structural constraints at multiple levels of granularity on the robustness of sentence compression at varying compression rates. Our evaluation demonstrates that the new feature set produces significantly improved compressions across a range of compression rates compared to existing state-of-the-art approaches. Thus, we name our system for generating compressions the Adjustable Rate Compressor (ARC).

Knight and Marcu (2000) (K&M, henceforth) presented two approaches to the sentence compression problem: one using a noisy channel model, the other using a decision-based model. The performances of the two models were comparable though their experiments suggested that the noisy channel model degraded more smoothly than the decision-based model when tested on out-of-domain data. Riezler et al. (2003) applied linguistically rich LFG grammars to a sentence compression system. Turner and Charniak (2005) achieved similar performance to K&M using an unsupervised approach that induced rules from the Penn Treebank.

A variety of feature encodings have previously been explored for the problem of sentence compression. Clarke and Lapata (2007) included discourse level features in their framework to leverage context for enhancing coherence. McDonald's (2006) model (M06, henceforth) is similar to K&M except that it uses discriminative online learning to train feature weights. A key aspect of the M06 approach is a decoding algorithm that searches the entire space of compressions using dynamic programming to choose the best compression (details in Section 2). We use M06 as a foundation for this work because its soft constraint approach allows for natural integration of additional classes of features. Similar to most previous approaches, our approach compresses sentences by deleting words only.

The remainder of the paper is organized as follows. Section 2 discusses the architectural framework. Section 3 describes the innovations in the proposed model. We conclude after presenting the results of our evaluation in Section 4.

2 Experimental Paradigm

Supervised approaches to sentence compression typically use parallel corpora consisting of original and compressed sentences (paired corpus, henceforth). In this paper, we will refer to these pairs as a 2-tuple $<x, y>$, where x is the original sentence and y is the compressed sentence.

We implemented the M06 system as an experimental framework in which to conduct our investigation. The system uses as input the paired corpus, the corresponding POS tagged corpus, the paired corpus parsed using the Charniak parser (Charniak, 2000), and dependency parses from the MST parser (McDonald et al., 2005). Features are extracted over adjacent pairs of words in the compressed sentence and weights are learnt at training time using the MIRA algorithm (Crammer and Singer, 2003). We decode as follows to find the best compression:

Let the score of a compression y for a sentence x be $s(x, y)$. This score is factored using a first-order Markov assumption over the words in the compressed sentence, and is defined by the dot product between a high dimensional feature representation and a corresponding weight vector (for details, refer to McDonald, 2006). The equations for decoding are as follows:

$$C[1] = 0.0$$
$$C[i] = \max_{j<i} C[j] + s(x, j, i), \forall i > 1$$

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 101–104,
Suntec, Singapore, 4 August 2009.

where C is the dynamic programming table and $C[i]$ represents the highest score for compressions ending at word i for the sentence x.

The M06 system takes the best scoring compression from the set of all possible compressions. In the ARC system, the model determines the compression rate and enforces a target compression length by altering the dynamic programming algorithm as suggested by M06:

$$C[1][1] = 0.0$$

$$C[1][r] = -\infty, \forall r > 1$$

$$\forall i > 1,$$

$$C[i][r] = \max_{j<i} C[j][r-1] + s(x, j, i)$$

where C is the dynamic programming table as before and $C[i][r]$ is the score for the best compression of length r that ends at position i in the sentence x. This algorithm runs in $O(n^2r)$ time.

We define the rate of human generated compressions in the training corpus as the gold standard compression rate (GSCR). We train a linear regression model over the training data to predict the GSCR for a sentence based on the ratio between the lengths of each compressed-original sentence pair in the training set. The predicted compression rate is used to force the system to compress sentences in the test set to a specific target length. Based on the computed regression, the formula for computing the Predicted Compression Rate (PCR) from the Original Sentence Length (OSL) is as follows:

$$PCR = 0.86 - 0.004 \times OSL$$

In our work, enforcing specific compression rates serves two purposes. First, it allows us to make a more controlled comparison across approaches, since variation in compression rate across approaches confounds comparison of other aspects of performance. Second, it allows us to investigate how alternative models work at higher compression rates. Here our primary contribution is of robustness of the approach with respect to alternative feature spaces and compression rates.

3 Extended Feature Set

A major focus of our work is the inclusion of new types of features derived from syntactic analyses in order to make the resulting compressions more grammatical and thus increase the versatility of the resulting compression models.

The M06 system uses features extracted from the POS tagged paired corpus: POS bigrams, POS context of the words added to or dropped from the compression, and other information about the dropped words. For a more detailed description, please refer to McDonald, 2006.

From the phrase structure trees, M06 extracts context information about nodes that subsume dropped words. These features attempt to approximately encode changes in the grammar rules between source and target sentences. Dependency features include information about the dropped words' parents as well as conjunction features of the word and the parent.

Our extensions to the M06 feature set are inspired by an analysis of the compressions generated by it, and allow for a richer encoding of dropped words and phrases using properties of the words and their syntactic relations to the rest of the sentence. Consider this example (dropped words are *marked as such*):

** 68000 Sweden AB of Uppsala , Sweden , introduced* the *TeleServe , an integrated* answering machine and voice-message handler that links a Macintosh to Touch-Tone *phones* .

Note in the above example that the syntactic head of the sentence *introduced* has been dropped. Using the dependency parse, we add a class of features to be learned during training that lets the system decide when to drop the syntactic head of the sentence. Also note that *answering machine* in the original sentence was preceded by *an* while the word *the* was used with *Teleserve* (dropped in the compression). While POS information helps the system to learn that *the answering machine* is a good POS sequence, we do not have information that links the correct article to the noun. Information from the dependency parse allows us to learn when we can drop words whose heads are retained and when we can drop a head and still retain the dependent.

Now, consider the following example:

Examples for editors are applicable to awk patterns , grep and egrep .

Here, *Examples* has been dropped, while *for editors* which has *Examples* as a head is retained. Besides, in the sequence, *editors are applicable…*, the word *editors* behaves as the subject of *are* although the correct compression would have *examples* as its subject. A change in the arguments of the verbs will distort the meaning of the sentence. We augmented the feature set to include a class of features about structural information that tells us when the subject (or object) of a verb can be dropped while the verb itself is retained. Thus, now if the system does retain the

are, it is more likely to retain the correct arguments of the word from the original sentence.

The new classes of features use only the dependency labels generated by the parser and are not lexicalized. Intuitively, these features help create units within the sentences that are tightly bound together, e.g., a subject and an object with its parent verb. We notice, as one would expect, that some dependency bindings are less strong than others. For instance, when faced with a choice, our system drops a relative pronoun thus breaking the dependency between the retained noun and the relative pronoun, rather than drop the noun, which was the retained subject.

Below is a summary of the information that the new features in our system encode:

[***Parent-Child***]- When a word is dropped, is its parent retained in the compression?

[***Dependent***]- When a word is dropped, are other words dependent on it (its children) also dropped or are they retained?

[***Verb-Arg***]- Information from the dependency parse about the subjects and objects of verbs can be used to encode more specific features (similar to the above) that say whether or not the subject (or object) was retained when the verb was dropped.

[***Sent-Head-Dep***]- Is the syntactic head of a sentence dropped?

4 Evaluation

We evaluate our model in comparison with M06. At training time, compression rates were not enforced on the ARC or M06 model. Our evaluation demonstrates that the proposed feature set produces more grammatical sentences across varying compression rates. In this section, GSCR denotes *gold standard compression rate* (i.e., the compression rate found in training data), CR denotes *compression rate*.

4.1 Corpora

Sentence compression systems have been tested on product review data from the Ziff-Davis (ZD, henceforth) Corpus by Knight and Marcu (2000), general news articles by Clarke and Lapata (CL, henceforth) corpus (2007) and biomedical articles (Lin and Wilbur, 2007). To evaluate our system, we used 2 test sets: **Set 1** contained 50 sentences; all 32 sentences from the ZD test set and 18 additional sentences chosen randomly from the CL test set; **Set 2** contained 40 sentences selected from the CL corpus, 20 of which were compressed at 75% of GSCR and 20 at 50% of GSCR (the percentages denote the enforced compression rates).

Three examples comparing compressed sentences are given below:

Original: *Like FaceLift, much of ATM 's screen performance depends on the underlying application.*
Human: *Much of ATM 's performance depends on the underlying application .*
M06: *'s screen performance depends on application*
ARC: *ATM 's screen performance depends on the underlying application .*

Original: *The discounted package for the Sparcserver 470 is priced at $89,900 , down from the regular $107,795 .*
Human: *The Sparcserver 470 is priced at $89,900 , down from the regular $107,795 .*
M06: *Sparcserver 470 is $89,900 regular $107,795*
ARC: *The discounted package is priced at $89,900 , regular $107,795 .*

The example below has compressions at 50% compression rate for M06 and ARC systems:

Original: *Cutbacks in local defence establishments is also a factor in some constituencies .*
M06: *establishments is a factor in some constituencies .*
ARC: *Cutbacks is a factor in some constituencies .*

Note that the subject of *is* is correctly retained in the ARC system.

4.2 User Study

In order to evaluate the effect of the features that we added to create the ARC model, we conducted a user study, adopting an experimental methodology similar to that used by K&M and M06. Each of four human judges, who were native speakers of English and not involved in the research we report in this paper, were instructed to rate two different sets of compressions along two dimensions, namely *Grammaticality* and *Completeness*, on a scale of 1 to 5. We chose to replace *Importance* (used by K&M), which is a task specific and possibly user specific notion, with the more general notion of *Completeness*, defined as the extent to which the compressed sentence is a complete sentence and communicates the main idea of the original sentence.

For Set 1, raters were given the original sentence and 4 compressed versions (presented in

random order as in the M06 evaluation): the human compression, the compression produced by the original M06 system, the compression from the M06 system with GSCR, and the ARC system with GSCR. For Set 2, raters were given the original sentence, this time with two compressed versions, one from the M06 system and one from the ARC system, which were presented in a random order. Table 1 presents all the results in terms of human ratings of Grammaticality and Completeness as well as automatically computed ROUGE F_1 scores (Lin and Hovy, 2003). The scores in parentheses denote standard deviations.

	Grammaticality (Human Scores)	Completeness (Human Scores)	ROUGE F_1
Gold Standard	4.60 (0.69)	3.80(.99)	1.00 (0)
ARC (GSCR)	3.70 (1.10)	3.50(1.10)	.72 (.18)
M06	3.50 (1.30)	3.10(1.30)	.70 (.20)
M06 (GSCR)	3.10 (1.10)	3.10(1.10)	.71 (.18)
ARC (75%CR)	2.60 (1.10)	2.60(1.10)	.72 (.14)
M06 (75%CR)	2.20 (1.20)	2.00(1.00)	.67 (.20)
ARC (50%CR)	2.30 (1.30)	1.90(1.00)	.54 (.22)
M06 (50%CR)	1.90 (1.10)	1.80(1.00)	.58 (.22)

Table 1: Results of human judgments and ROUGE F_1

ROUGE scores were determined to have a significant positive correlation both with Grammaticality ($R = .46$, $p < .0001$) and Completeness ($R = .39$, $p < .0001$) when averaging across the 4 judges' ratings. On Set 1, a 2-tailed paired *t*-test reveals similar patterns for Grammaticality and Completeness: the human compressions are significantly better than any of the systems. ARC is significantly better than M06, both with enforced GSCR and without. M06 without GSCR is significantly better than M06 with GSCR. In Set 2 (with 75% and 50% GSCR enforced), the quality of compressions degrade as compression rate is made more severe; however, the ARC model consistently outperforms the M06 model with a statistically significant margin across compression rates on both evaluation criteria.

5 Conclusions and Future Work

In this paper, we designed a set of new classes of features to generate better compressions, and they were found to produce statistically significant improvements over the state-of-the-art. However, although the user study demonstrates the expected positive impact of grammatical features, an error analysis (Gupta et al., 2009) reveals some limitations to improvements that can be obtained using grammatical features that refer only to the source sentence structure, since the syntax of the source sentence is frequently not preserved in the gold standard compression. In our future work, we hope to explore alternative approaches that allow reordering or paraphrasing along with deleting words to make compressed sentences more grammatical and coherent.

Acknowledgments

The authors thank Kevin Knight and Daniel Marcu for sharing the Ziff-Davis corpus as well as the output of their systems, and the anonymous reviewers for their comments. This work was supported by the Cognitive and Neural Sciences Division, grant number N00014-00-1-0600.

References

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. of NAACL*.

James Clarke and Mirella Lapata, 2007. Modelling Compression With Discourse Constraints. In *Proc. of EMNLP-CoNLL*.

Koby Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multi-class problems. *JMLR.*

Naman K. Gupta, Sourish Chaudhuri and Carolyn P. Rosé, 2009. Evaluating the Syntactic Transformations in Gold Standard Corpora for Statistical Sentence Compression . In *Proc. of HLT-NAACL*.

Kevin Knight and Daniel Marcu. 2000. Statistics-Based Summarization – Step One: Sentence Compression. In *Proc. of AAAI*.

Jimmy Lin and W. John Wilbur. 2007. Syntactic sentence compression in the biomedical domain: facilitating access to related articles. *Information Retrieval*, 10(4):393-414.

Chin-Yew Lin and Eduard H. Hovy 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *Proc. of HLT-NAACL*.

Ryan McDonald, 2006. Discriminative sentence compression with soft syntactic constraints. In *Proc. of EACL*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proc.of ACL*.

S. Riezler, T. H. King, R. Crouch, and A. Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proc. of HLT-NAACL*.

Query-Focused Summaries or Query-Biased Summaries ?

Rahul Katragadda
Language Technologies Research Center
IIIT Hyderabad
rahul_k@research.iiit.ac.in

Vasudeva Varma
Language Technologies Research Center
IIIT Hyderabad
vv@iiit.ac.in

Abstract

In the context of the Document Understanding Conferences, the task of Query-Focused Multi-Document Summarization is intended to improve agreement in content among human-generated model summaries. *Query-focus* also aids the automated summarizers in directing the summary at specific topics, which may result in better agreement with these model summaries. However, while query focus correlates with performance, we show that high-performing automatic systems produce summaries with disproportionally higher query term density than human summarizers do. Experimental evidence suggests that automatic systems heavily rely on query term occurrence and repetition to achieve good performance.

1 Introduction

The problem of automatically summarizing text documents has received a lot of attention since the early work by Luhn (Luhn, 1958). Most of the current automatic summarization systems rely on a sentence extractive paradigm, where key sentences in the original text are selected to form the summary based on the clues (or heuristics), or learning based approaches.

Common approaches for identifying key sentences include: training a binary classifier (Kupiec et al., 1995), training a Markov model or CRF (Conroy et al., 2004; Shen et al., 2007) or directly assigning weights to sentences based on a variety of features and heuristically determined feature weights (Toutanova et al., 2007). But, the question of which components and features of automatic summarizers contribute most to their performance has largely remained unanswered (Marcu and Gerber, 2001), until Nenkova et al. (Nenkova et al., 2006) explored the contribution of frequency based measures. In this paper, we examine the role a *query* plays in automated multi-document summarization of newswire.

One of the issues studied since the inception of automatic summarization is that of human agreement: different people choose different content for their summaries (Rath et al., 1961; van Halteren and Teufel, 2003; Nenkova et al., 2007). Later, it was assumed (Dang, 2005) that having a question/query to provide focus would improve agreement between any two human-generated model summaries, as well as between a model summary and an automated summary. Starting in 2005 until 2007, a *query-focused multi-document summarization* task was conducted as part of the annual Document Understanding Conference. This task models a real-world complex question answering scenario, where systems need to synthesize from a set of 25 documents, a brief (250 words), well organized fluent answer to an information need.

Query-focused summarization is a topic of ongoing importance within the summarization and question answering communities. Most of the work in this area has been conducted under the guise of "*query-focused multi-document summarization*", "*descriptive question answering*", or even "*complex question answering*".

In this paper, based on structured empirical evaluations, we show that most of the systems participating in DUC's Query-Focused Multi-Document Summarization (QF-MDS) task have been query-biased in building extractive summaries. Throughout our discussion, the term *'query-bias'*, with respect to a sentence, is precisely defined to mean that the sentence has at least one query term within it. The term *'query-focus'* is less precisely defined, but is related to the cognitive task of focusing a summary on the query, which we assume humans do naturally. In other words, the human generated model summaries are assumed to be query-focused.

Here we first discuss *query-biased* content in Summary Content Units (SCUs) in Section 2 and then in Section 3 by building formal models on *query-bias* we discuss why/how automated systems are query-biased rather than being query-focused.

2 Query-biased content in Summary Content Units (SCUs)

Summary content units, referred as SCUs hereafter, are semantically motivated subsentential units that are variable in length but not bigger than a sentential clause. SCUs are constructed from annotation of a collection of human summaries on a given document collection. They are identified by noting information that is repeated across summaries. The repetition is as small as a modifier of a noun phrase or as large as a clause. The evaluation method that is based on overlapping SCUs in human and automatic summaries is called the

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 105–108,
Suntec, Singapore, 4 August 2009.

<document name="APW20000824.0204">
<line>*A lawyer who specializes in bankrupting hate groups is going after the Aryan Nations, whose compound in the Idaho woods has served as a clubhouse for some of America's most violent racists.*</line>
<line>*In a lawsuit that goes to trial Monday, attorney Morris Dees of the Southern Poverty Law Center is representing a mother and son who were attacked by security guards for the white supremacist group.*
<annotation scu-count="1" sum-count="8" sums="13,14,15,23,24,29,30,9">
<scu uid="24" label="SPLC takes legal action against civil rights abuses" weight="3"/></annotation></line>
<line>*The victims are suing the Aryan Nations and founder Richard Butler.*
<annotation scu-count="0" sum-count="1" sums="29"/></line>

Figure 1: SCU annotation of a source document.

pyramid method (Nenkova et al., 2007).

The University of Ottawa has organized the pyramid annotation data such that for some of the sentences in the original document collection, a list of corresponding content units is known (Copeck et al., 2006). A sample of an SCU mapping from topic *D0701A* of the DUC 2007 QF-MDS corpus is shown in Figure 1. Three sentences are seen in the figure among which two have been annotated with system IDs and SCU weights wherever applicable. The first sentence has not been picked by any of the summarizers participating in Pyramid Evaluations, hence it is unknown if the sentence would have contributed to any SCU. The second sentence was picked by 8 summarizers and that sentence contributed to an SCU of weight 3. The third sentence in the example was picked by one summarizer, however, it did not contribute to any SCU. This example shows all the three types of sentences available in the corpus: unknown samples, positive samples and negative samples.

We extracted the positive and negative samples in the source documents from these annotations; types of second and third sentences shown in Figure 1. A total of 14.8% sentences were annotated to be either positive or negative. When we analyzed the positive set, we found that 84.63% sentences in this set were *query-biased*. Also, on the negative sample set, we found that 69.12% sentences were *query-biased*. That is, on an average, 76.67% of the sentences picked by any automated summarizer are *query-biased*. On the other hand, for human summaries only 58% sentences were *query-biased*. All the above numbers are based on the DUC 2007 dataset shown in **boldface** in Table 1 [1].

There is one caveat: The annotated sentences come only from the summaries of systems that participated in the pyramid evaluations. Since only 13 among a total 32 participating systems were evaluated using pyramid evaluations, the dataset is limited. However, despite this small issue, it is very clear that at least those systems that participated in pyramid evaluations have been biased towards query-terms, or at least, they have been better at correctly identifying important sentences from the query-biased sentences than from query-unbiased sentences.

[1]We used DUC 2007 dataset for all experiments reported.

3 Formalizing *query-bias*

Our search for a formal method to capture the relation between occurrence of query-biased sentences in the input and in summaries resulted in building binomial and multinomial model distributions. The distributions estimated were then used to obtain the likelihood of a query-biased sentence being emitted into a summary by each system.

For the DUC 2007 data, there were 45 summaries for each of the 32 systems (labeled 1-32) among which 2 were baselines (labeled 1 and 2), and 18 summaries from each of 10 human summarizers (labeled A-J). We computed the log-likelihood, $\log$*(L[summary;p(*C_i*)])*, of all human and machine summaries from DUC'07 query focused multi-document summarization task, based on both distributions described below (see Sections 3.1, 3.2).

3.1 The binomial model

We represent the set of sentences as a binomial distribution over type of sentences. Let C_0 and C_1 denote the sets of sentences without and with query-bias respectively. Let p(C_i) be the probability of emitting a sentence from a specified set. It is also obvious that query-biased sentences will be assigned lower emission probabilities, because the occurrence of query-biased sentences in the input is less likely. On average each topic has 549 sentences, among which 196 contain a query term; which means only 35.6% sentences in the input were query-biased. Hence, the likelihood function here denotes the likelihood of a summary to contain non query-biased sentences. Humans' and systems' summaries must now constitute low likelihood to show that they rely on *query-bias*.

The likelihood of a summary then is :

$$L[summary; p(C_i)] = \frac{N!}{n_0!n_1!} p(C_0)^{n_0} p(C_1)^{n_1} \quad (1)$$

Where N is the number of sentences in the summary, and n_0 + n_1 = N; n_0 and n_1 are the cardinalities of C_0 and C_1 in the summary. Table 2 shows various systems with their ranks based on ROUGE-2 and the average log-likelihood scores. The ROUGE (Lin, 2004) suite of metrics are n-gram overlap based metrics that have been shown to highly correlate with human evaluations on content responsiveness. ROUGE-2 and ROUGE-SU4 are the official ROUGE metrics for evaluating *query-focused multi-document summarization* task since DUC 2005.

3.2 The multinomial model

In the previous section (Section 3.1), we described the binomial model where we classified each sentence as being *query-biased* or not. However, if we were to quantify the amount of *query-bias* in a sentence, we associate each sentence to one among k possible classes leading to a multinomial distribution. Let $C_i \in$

Dataset	total	positive	biased positive	negative	biased negative	% bias in positive	% bias in negative
DUC 2005	24831	1480	1127	1912	1063	76.15	55.60
DUC 2006	14747	1047	902	1407	908	86.15	71.64
DUC 2007	***12832***	***924***	***782***	***975***	***674***	***84.63***	***69.12***

Table 1: Statistical information on counts of *query-biased* sentences.

ID	rank	LL	ROUGE-2	ID	rank	LL	ROUGE-2	ID	rank	LL	ROUGE-2
1	31	-1.9842	0.06039	***J***		***-3.9465***	0.13904	24	4	-5.8451	0.11793
C		-2.1387	0.15055	***E***		***-3.9485***	0.13850	9	12	-5.9049	0.10370
16	32	-2.2906	0.03813	10	28	-4.0723	0.07908	14	14	-5.9860	0.10277
27	30	-2.4012	0.06238	21	22	-4.2460	0.08989	5	23	-6.0464	0.08784
6	29	-2.5536	0.07135	G		-4.3143	0.13390	4	3	-6.2347	0.11887
12	25	-2.9415	0.08505	25	27	-4.4542	0.08039	20	6	-6.3923	0.10879
I		-3.0196	0.13621	B		-4.4655	0.13992	29	2	-6.4076	0.12028
11	24	-3.0495	0.08678	19	26	-4.6785	0.08453	3	9	-7.1720	0.10660
28	16	-3.1932	0.09858	26	21	-4.7658	0.08989	8	11	-7.4125	0.10408
2	18	-3.2058	0.09382	23	7	-5.3418	0.10810	17	15	-7.4458	0.10212
D		***-3.2357***	0.17528	30	10	-5.4039	0.10614	13	5	-7.7504	0.11172
H		***-3.4494***	0.13001	7	8	-5.6291	0.10795	32	17	-8.0117	0.09750
A		***-3.6481***	0.13254	18	19	-5.6397	0.09170	22	13	-8.9843	0.10329
F		***-3.8316***	0.13395	15	1	-5.7938	0.12448	31	20	-9.0806	0.09126

Table 2: Rank, Averaged log-likelihood score based on **binomial model**, true ROUGE-2 score for the summaries of various systems in DUC'07 *query-focused multi-document summarization* task.

ID	rank	LL	ROUGE-2	ID	rank	LL	ROUGE-2	ID	rank	LL	ROUGE-2
1	31	-4.6770	0.06039	10	28	-8.5004	0.07908	5	23	-14.3259	0.08784
16	32	-4.7390	0.03813	G		-9.5593	0.13390	9	12	-14.4732	0.10370
6	29	-5.4809	0.07135	***E***		***-9.6831***	0.13850	22	13	-14.8557	0.10329
27	30	-5.5110	0.06238	26	21	-9.7163	0.08989	4	3	-14.9307	0.11887
I		-6.7662	0.13621	***J***		***-9.8386***	0.13904	18	19	-15.0114	0.09170
12	25	-6.8631	0.08505	19	26	-10.3226	0.08453	14	14	-15.4863	0.10277
2	18	-6.9363	0.09382	B		-10.4152	0.13992	20	6	-15.8697	0.10879
C		-7.2497	0.15055	25	27	-10.7693	0.08039	32	17	-15.9318	0.09750
H		***-7.6657***	0.13001	29	2	-12.7595	0.12028	7	8	-15.9927	0.10795
11	24	-7.8048	0.08678	21	22	-13.1686	0.08989	17	15	-17.3737	0.10212
A		***-7.8690***	0.13254	24	4	-13.2842	0.11793	8	11	-17.4454	0.10408
D		***-8.0266***	0.17528	30	10	-13.3632	0.10614	31	20	-17.5615	0.09126
28	16	-8.0307	0.09858	23	7	-13.7781	0.10810	3	9	-19.0495	0.10660
F		***-8.2633***	0.13395	15	1	-14.2832	0.12448	13	5	-19.3089	0.11172

Table 3: Rank, Averaged log-likelihood score based on **multinomial model**, true ROUGE-2 score for the summaries of various systems in DUC'07 *query-focused multi-document summarization* task.

$\{C_0, C_1, C_2, \ldots, C_k\}$ denote the k levels of *query-bias*. C_i is the set of sentences, each having i query terms.

The number of sentences participating in each class varies highly, with C_0 bagging a high percentage of sentences (64.4%) and the rest $\{C_1, C_2, \ldots, C_k\}$ distributing among themselves the rest 35.6% sentences. Since the distribution is highly-skewed, distinguishing systems based on log-likelihood scores using this model is easier and perhaps more accurate. Like before, Humans' and systems' summaries must now constitute low likelihood to show that they rely on *query-bias*.

The likelihood of a summary then is :

$$L[summary; p(C_i)] = \frac{N!}{n_0! n_1! \cdots n_k!} p(C_0)^{n_0} p(C_1)^{n_1} \cdots p(C_k)^{n_k} \quad (2)$$

Where N is the number of sentences in the summary, and $n_0 + n_1 + \cdots + n_k = N$; $n_0, n_1, \cdots, n_k$ are respectively the cardinalities of $C_0, C_1, \cdots, C_k$, in the summary. Table 3 shows various systems with their ranks based on ROUGE-2 and the average log-likelihood scores.

3.3 Correlation of ROUGE and log-likelihood scores

Tables 2 and 3 display log-likelihood scores of various systems in the descending order of log-likelihood scores along with their respective ROUGE-2 scores. We computed the pearson correlation coefficient (ρ) of 'ROUGE-2 and log-likelihood' and 'ROUGE-SU4 and log-likelihood'. This was computed for systems (ID: *1-32*) (*r1*) and for humans (ID: *A-J*) (*r2*) separately, and for both distributions.

For the binomial model, *r1* = -0.66 and *r2* = 0.39 was obtained. This clearly indicates that there is a strong negative correlation between likelihood of occurrence of a non-query-term and ROUGE-2 score. That is, a strong positive correlation between likelihood of occur-

rence of a query-term and ROUGE-2 score. Similarly, for human summarizers there is a weak negative correlation between likelihood of occurrence of a query-term and ROUGE-2 score. The same correlation analysis applies to ROUGE-SU4 scores: *r1* = -0.66 and *r2* = 0.38.

Similar analysis with the multinomial model have been reported in Tables 4 and 5. Tables 4 and 5 show the correlation among ROUGE-2 and log-likelihood scores for systems[2] and humans[3].

ρ	**ROUGE-2**	**ROUGE-SU4**
binomial	-0.66	-0.66
multinomial	-0.73	-0.73

Table 4: Correlation of ROUGE measures with log-likelihood scores for automated systems

ρ	**ROUGE-2**	**ROUGE-SU4**
binomial	0.39	0.38
multinomial	0.15	0.09

Table 5: Correlation of ROUGE measures with log-likelihood scores for humans

4 Conclusions and Discussion

Our results underscore the differences between human and machine generated summaries. Based on Summary Content Unit (SCU) level analysis of query-bias we argue that most systems are better at finding important sentences only from *query-biased* sentences. More importantly, we show that on an average, 76.67% of the sentences picked by any automated summarizer are *query-biased*. When asked to produce query-focused summaries, humans do not rely to the same extent on the repetition of query terms.

We further confirm based on the likelihood of emitting non query-biased sentence, that there is a strong (negative) correlation among systems' likelihood score and ROUGE score, which suggests that systems are trying to improve performance based on ROUGE metrics by being biased towards the *query terms*. On the other hand, humans do not rely on *query-bias*, though we do not have statistically significant evidence to suggest it. We have also speculated that the multinomial model helps in better capturing the variance across the systems since it distinguishes among *query-biased* sentences by quantifying the amount of query-bias.

From our point of view, most of the extractive summarization algorithms are formalized based on a bag-of-words query model. The innovation with individual approaches has been in formulating the actual algorithm on top of the query model. We speculate that the real difference in human summarizers and automated summarizers could be in the way a query (or relevance) is represented. Traditional query models from IR literature have been used in summarization research thus far, and though some previous work (Amini and Usunier, 2007) tries to address this issue using contextual query expansion, new models to represent the query is perhaps one way to induce topic-focus on the summary. IR-like query models, which are designed to handle 'short keyword queries', are perhaps not capable of handling 'an elaborate query' in case of summarization. Since the notion of *query-focus* is apparently missing in any or all of the algorithms, the future summarization algorithms must try to incorporate this while designing new algorithms.

[2]All the results in Table 4 are statistically significant with p-value ($p < 0.00004$, N=32)

[3]None of the results in Table 5 are statistically significant with p-value ($p > 0.265$, N=10)

Acknowledgements

We thank Dr Charles L A Clarke at the University of Waterloo for his deep reviews and discussions on earlier versions of the paper. We are also grateful to all the anonymous reviewers for their valuable comments.

References

Massih R. Amini and Nicolas Usunier. 2007. A contextual query expansion approach by term clustering for robust text summarization. In *the proceedings of Document Understanding Conference*.

John M. Conroy, Judith D. Schlesinger, Jade Goldstein, and Dianne P. O'leary. 2004. Left-brain/right-brain multi-document summarization. In *the proceedings of Document Understanding Conference (DUC) 2004*.

Terry Copeck, D Inkpen, Anna Kazantseva, A Kennedy, D Kipp, Vivi Nastase, and Stan Szpakowicz. 2006. Leveraging duc. In *proceedings of DUC 2006*.

Hoa Trang Dang. 2005. Overview of duc 2005. In *proceedings of Document Understanding Conference*.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *the proceedings of ACM SIGIR'95*, pages 68–73. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *the proceedings of ACL Workshop on Text Summarization Branches Out*. ACL.

H.P. Luhn. 1958. The automatic creation of literature abstracts. In *IBM Journal of Research and Development, Vol. 2, No. 2, pp. 159-165, April 1958.*

Daniel Marcu and Laurie Gerber. 2001. An inquiry into the nature of multidocument abstracts, extracts, and their evaluation. In *Proceedings of the NAACL-2001 Workshop on Automatic Summarization*.

Ani Nenkova, Lucy Vanderwende, and Kathleen McKeown. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 573–580, New York, NY, USA. ACM.

Ani Nenkova, Rebecca Passonneau, and Kathleen McKeown. 2007. The pyramid method: Incorporating human content selection variation in summarization evaluation. In *ACM Trans. Speech Lang. Process.*, volume 4, New York, NY, USA. ACM.

G.J. Rath, A. Resnick, and R. Savage. 1961. The formation of abstracts by the selection of sentences: Part 1: Sentence selection by man and machines. In *Journal of American Documentation.*, pages 139–208.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document summarization using conditional random fields. In *the proceedings of IJCAI '07.*, pages 2862–2867. IJCAI.

Kristina Toutanova, Chris Brockett, Michael Gamon, Jagadeesh Jagarlamundi, Hisami Suzuki, and Lucy Vanderwende. 2007. The pythy summarization system: Microsoft research at duc 2007. In *the proceedings of Document Understanding Conference*.

Hans van Halteren and Simone Teufel. 2003. Examining the consensus between human summaries: initial experiments with factoid analysis. In *HLT-NAACL 03 Text summarization workshop*, pages 57–64, Morristown, NJ, USA. Association for Computational Linguistics.

Using Generation for Grammar Analysis and Error Detection

Michael Wayne Goodman*
University of Washington
Dept. of Linguistics
Box 354340 Seattle, WA 98195, USA
goodmami@u.washington.edu

Francis Bond
NICT Language Infrastructure Group
3-5 Hikaridai, Seika-cho, Sōraku-gun,
Kyoto, 619-0289 Japan
bond@ieee.org

Abstract

We demonstrate that the bidirectionality of deep grammars, allowing them to generate as well as parse sentences, can be used to automatically and effectively identify errors in the grammars. The system is tested on two implemented HPSG grammars: Jacy for Japanese, and the ERG for English. Using this system, we were able to increase generation coverage in Jacy by 18% (45% to 63%) with only four weeks of grammar development.

1 Introduction

Linguistically motivated analysis of text provides much useful information for subsequent processing. However, this is generally at the cost of reduced coverage, due both to the difficulty of providing analyses for all phenomena, and the complexity of implementing these analyses. In this paper we present a method of identifying problems in a deep grammar by exploiting the fact that it can be used for both parsing (interpreting text into semantics) and generation (realizing semantics as text). Since both parsing and generation use the same grammar, their performance is closely related: in general improving the performance or cover of one direction will also improve the other. (Flickinger, 2008)

The central idea is that we test the grammar on a full round trip: parsing text to its semantic representation and then generating from it. In general, any sentence where we cannot reproduce the original, or where the generated sentence significantly differs from the original, identifies a flaw in the grammar, and with enough examples we can pinpoint the grammar rules causing these problems. We call our system **Egad**, which stands for Erroneous Generation Analysis and Detection.

*This research was carried out while visiting NICT.

2 Background

This work was inspired by the error mining approach of van Noord (2004), who identified problematic input for a grammar by comparing sentences that parsed and those that didn't from a large corpus. Our approach takes this idea and further applies it to generation. We were also inspired by the work of Dickinson and Lee (2008), whose "variation n-gram method" models the likelihood a particular argument structure (semantic annotation) is accurate given the verb and some context.

We tested **Egad** on two grammars: Jacy (Siegel, 2000), a Japanese grammar and the English Resource Grammar (ERG) (Flickinger, 2000, 2008) from the DELPH-IN[1] group. Both grammars are written in the Head-driven Phrase Structure Grammar (HPSG) (Pollard and Sag, 1994) framework, and use Minimal Recursion Semantics (MRS) (Copestake et al., 2005) for their semantic representations. The Tanaka Corpus (Tanaka, 2001) provides us with English and Japanese sentences.

The specific motivation for this work was to increase the quality and coverage of generated paraphrases using Jacy and the ERG. Bond et al. (2008) showed they could improve the performance of a statistical machine translation system by training on a corpus that included paraphrased variations of the English text. We want to do the same with Japanese text, but Jacy was not able to produce paraphrases as well (the ERG had 83% generation coverage, while Jacy had 45%) Improving generation would also greatly benefit X-to-Japanese machine translation tasks using Jacy.

2.1 Concerning Grammar Performance

There is a difference between the theoretical and practical power of the grammars. Sometimes the

[1]Deep Linguistic Processing with HPSG Initiative – see http://www.delph-in.net for background information, including the list of current participants and pointers to available resources and documentation

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 109–112,
Suntec, Singapore, 4 August 2009.

parser or generator can reach the memory (i.e. edge) limit, resulting in a valid result not being returned. Also, we only look at the top-ranked[2] parse and the first five generations for each item. This is usually not a problem, but it could cause **Egad** to report false positives.

HPSG grammars are theoretically symmetric between parsing and generation, but in practice this is not always true. For example, to improve performance, semantically empty lexemes are not inserted into a generation unless a "trigger-rule" defines a context for them. These trigger-rules may not cover all cases.

3 Grammar Analysis

When analyzing a grammar, **Egad** looks at all input sentences, parses, and generations processed by the grammar and uses the information therein to determine characteristics of these items. These characteristics are encoded in a vector that can be used for labeling and searching items. Some characteristics are useful for error mining, while others are used for grammar analysis.

3.1 Characteristic Types

Egad determines both general characteristics of an item (parsability and generability), and characteristics comparing parses with generations.

General characteristics show whether each item could: be parsed ("parsable"), generate from parsed semantics ("generable"), generate the original parsed sentence ("reproducible"), and generate other sentences ("paraphrasable").

For comparative characteristics, **Egad** compares every generated sentence to the parsed sentence whence its semantics originated, and determines if the generated sentence uses the same set of lexemes, derivation tree,[3] set of rules, surface form, and MRS as the original.

3.2 Characteristic Patterns

Having determined all applicable characteristics for an item or a generated sentence, we encode the values of those characteristics into a vector. We call this vector a **characteristic pattern**, or CP. An example CP showing general characteristics is:

```
0010 -----
```

The first four digits are read as: the item is parsable, generable, not reproducible, and is paraphrasable. The five following dashes are for comparative characteristics and are inapplicable except for generations.

3.3 Utility of Characteristics

Not all characteristics are useful for all tasks. We were interested in improving Jacy's ability to generate sentences, so we primarily looked at items that were parsable but ungenerable. In comparing generated sentences with the original parsed sentence, those with differing semantics often point to errors, as do those with a different surface form but the same derivation tree and lexemes (which usually means an inflectional rule was misapplied).

4 Problematic Rule Detection

Our method for detecting problematic rules is to train a maximum entropy-based classifier[4] with n-gram paths of rules from a derivation tree as features and characteristic patterns as labels. Once trained, we do feature-selection to look at what paths of rules are most predictive of certain labels.

4.1 Rule Paths

We extract n-grams over **rule paths**, or RPs, which are downward paths along the derivation tree. (Toutanova et al., 2005) By creating separate RPs for each branch in the derivation tree, we retain some information about the order of rule application without overfitting to specific tree structures. For example, Figure 1 is the derivation tree for (1). A couple of RPs extracted from the derivation tree are shown in Figure 2.

(1) 写真 写り が いい
shashin-utsuri-ga ii
picture-taking-NOM good
(X is) good at taking pictures.

4.2 Building a Model

We build a classification model by using a parsed or generated sentence's RPs as features and that sentence's CP as a label. The set of RPs includes n-grams over all specified values of N. The labels are, to be more accurate, regular expressions of

[2] Jacy and the ERG both have parse-ranking models.

[3] In comparing the derivation trees, we only look at phrasal nodes. Lexemes and surface forms are not compared.

[4] We would like to look at using different classifiers here, such as Decision Trees. We initially chose MaxEnt because it was easy to implement, and have since had little motivation to change it because it produced useful results.

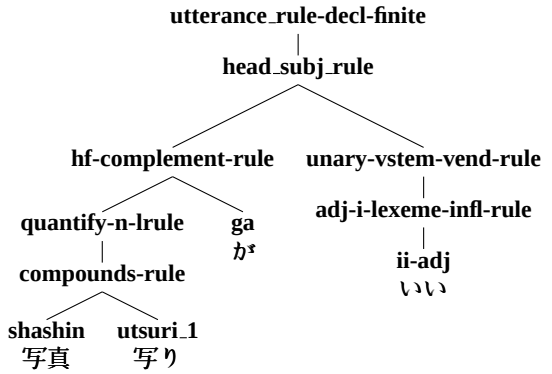


Figure 1: Derivation tree for (1)

quantify-n-lrule → *compounds-rule* → *shashin*
quantify-n-lrule → *compounds-rule* → *utsuri_1*

Figure 2: Example RPs extracted from Figure 1

CPs and may be fully specified to a unique CP or generalize over several.[5] The user can weight the RPs by their N value (e.g. to target unigrams).

4.3 Finding Problematic Rules

After training the model, we have a classifier that predicts CPs given a set of RPs. What we want, however, is the RP most strongly associated with a given CP. The classifier we use provides an easy method to get the score a given feature has for some label. We iterate over all RPs, get their score, then sort them based on the score. To help eliminate redundant results, we exclude any RP that either subsumes or is subsumed by a previous (i.e. higher ranked) RP.

Given a CP, the RP with the highest score should indeed be the one most closely associated to that CP, but it might not lead to the greatest number of items affected. Fixing the second highest ranked RP, for example, may improve more items than fixing the top ranked one. To help the grammar developer decide the priority of problems to fix, we also output the count of items observed with the given CP and RP.

5 Results and Evaluation

We can look at two sets of results: how well **Egad** was able to analyze a grammar and detect errors, and how well a grammar developer could use **Egad** to fix a problematic grammar. While the latter is also influenced by the skill of the grammar developer, we are interested in how well **Egad** points to the most significant errors, and how it can help reduce development time.

5.1 Error Mining

Table 1 lists the ten highest ranked RPs associated with items that could parse but could not generate in Jacy. Some RPs appear several times in different contexts. We made an effort to decrease the redundancy, but clearly this could be improved.

From this list of ten problematic RPs, there are four unique problems: quantify-n-lrule (noun quantification), no-nspec (noun specification), to-comp-quotarg (と *to* quotative particle), and te-adjunct (verb conjugation). The extra rules listed in each RP show the context in which each problem occurs, and this can be informative as well. For instance, *quantify-n-lrule* occurs in two primary contexts (above *compounds-rule* and *nominal-numcl-rule*). The symptoms of the problem occur in the interation of rules in each context, but the source of the problem is *quantify-n-lrule*.

Further, the problems identified are not always lexically marked. *quantify-n-lrule* occurs for all bare noun phrases (ie. without determiners). This kind of error cannot be accurately identified by using just word or POS n-grams, we need to use the actual parse tree.

5.2 Error Correction

Egad greatly facilitated our efforts to find and fix a wide variety of errors in Jacy. For example, we restructured semantic predicate hierarchies, fixed noun quantification, allowed some semantically empty lexemes to generate in certain contexts, added pragmatic information to distinguish between politeness levels in pronouns, allowed imperatives to generate, allowed more constructions for numeral classifiers, and more.

Egad also identified some issues with the ERG: both over-generation (an under-constrained inflectional rule) and under-generation (sentences with the construction *take* $\{care|charge|\ldots\}$ *of* were not generating).

5.3 Updated Grammar Statistics

After fixing the most significant problems in Jacy (outlined in Section 5.2) as reported by **Egad**, we obtained new statistics about the grammar's coverage and characteristics. Table 2 shows the original and updated general statistics for Jacy. We increased generability by 18%, doubled reproducibility, and increased paraphrasability by 17%.

[5] For example, `/0010 -----/` is fully specified. `/00.. -----/` marginalizes two general characteristics

Score	Count	Rule Path N-grams
1.42340952569648	109	hf-complement-rule → quantify-n-lrule → compounds-rule
0.960090299833317	54	hf-complement-rule → quantify-n-lrule → nominal-numcl-rule → head-specifier-rule
0.756227560530811	63	head-specifier-rule → hf-complement-rule → no-nspec → "の"
0.739668926140179	62	hf-complement-rule → head-specifier-rule → hf-complement-rule → no-nspec
0.739090261637851	22	hf-complement-rule → hf-adj-i-rule → quantify-n-lrule → compounds-rule
0.694215264789286	36	hf-complement-rule → hf-complement-rule → to-comp-quotarg → "と"
0.676244980660372	82	vstem-vend-rule → te-adjunct → "て"
0.617621482523537	26	hf-complement-rule → hf-complement-rule → to-comp-varg → "と"
0.592260546433334	36	hf-adj-i-rule → hf-complement-rule → quantify-n-lrule → nominal-numcl-rule
0.564790702894285	62	quantify-n-lrule → compounds-rule → vn2n-det-lrule

Table 1: Top 10 RPs for ungenerable items

	Original	Modified
Parsable	82%	83%
Generable	45%	63%
Reproducible	11%	22%
Paraphrasable	44%	61%

Table 2: Jacy's improved general statistics

As an added bonus, our work focused on improving generation also improved parsability by 1%. Work is now continuing on fixing the remainder of the identified errors.

6 Future Work

In future iterations of **Egad**, we would like to expand our feature set (e.g. information from failed parses), and make the system more robust, such as replacing lexical-ids (specific to a lexeme) with lexical-types, since all lexemes of the same type should behave identically. A more long-term goal would allow **Egad** to analyze the internals of the grammar and point out specific features within the grammar rules that are causing problems. Some of the errors detected by **Egad** have simple fixes, and we believe there is room to explore methods of automatic error correction.

7 Conclusion

We have introduced a system that identifies errors in implemented HPSG grammars, and further finds and ranks the possible sources of those problems. This tool can greatly reduce the amount of time a grammar developer would spend finding bugs, and helps them make informed decisions about which bugs are best to fix. In effect, we are substituting cheap CPU time for expensive grammar developer time. Using our system, we were able to improve Jacy's absolute generation coverage by 18% (45% to 63%) with only four weeks of grammar development.

8 Acknowledgments

Thanks to NICT for their support, Takayuki Kuribayashi for providing native judgments, and Marcus Dickinson for comments on an early draft.

References

Francis Bond, Eric Nichols, Darren Scott Appling, and Michael Paul. 2008. Improving statistical machine translation by paraphrasing the training data. In *International Workshop on Spoken Language Translation*, pages 150–157. Honolulu.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal Recursion Semantics. An introduction. *Research on Language and Computation*, 3(4):281–332.

Markus Dickinson and Chong Min Lee. 2008. Detecting errors in semantic annotation. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28. (Special Issue on Efficient Processing with HPSG).

Dan Flickinger. 2008. The English resource grammar. Technical Report 2007-7, LOGON, `http://www.emmtee.net/reports/7.pdf`. (Draft of 2008-11-30).

Carl Pollard and Ivan A. Sag. 1994. *Head Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Melanie Siegel. 2000. HPSG analysis of Japanese. In Wolfgang Wahlster, editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 265 – 280. Springer, Berlin, Germany.

Yasuhito Tanaka. 2001. Compilation of a multilingual parallel corpus. In *Proceedings of PACLING 2001*, pages 265–268. Kyushu. (`http://www.colips.org/afnlp/archives/pacling2001/pdf/tanaka.pdf`).

Kristina Toutanova, Christopher D. Manning, Dan Flickinger, and Stephan Oepen. 2005. Stochastic HPSG parse disambiguation using the redwoods corpus. *Research on Language and Computation*, 3(1):83–105.

Gertjan van Noord. 2004. Error mining for wide-coverage grammar engineering. In *42nd Annual Meeting of the Association for Computational Linguistics: ACL-2004*. Barcelona.

An Integrated Multi-document Summarization Approach based on Word Hierarchical Representation

You Ouyang, Wenji Li, Qin Lu
Department of Computing
The Hong Kong Polytechnic University
{csyouyang,cswjli,csluqin}@comp.polyu.edu.hk

Abstract

This paper introduces a novel hierarchical summarization approach for automatic multi-document summarization. By creating a hierarchical representation of the words in the input document set, the proposed approach is able to incorporate various objectives of multi-document summarization through an integrated framework. The evaluation is conducted on the DUC 2007 data set.

1 Introduction and Background

Multi-document summarization requires creating a short summary from a set of documents which concentrate on the same topic. Sometimes an additional query is also given to specify the information need of the summary. Generally, an effective summary should be relevant, concise and fluent. It means that the summary should cover the most important concepts in the original document set, contain less redundant information and should be well-organized.

Currently, most successful multi-document summarization systems follow the extractive summarization framework. These systems first rank all the sentences in the original document set and then select the most salient sentences to compose summaries for a good coverage of the concepts. For the purpose of creating more concise and fluent summaries, some intensive post-processing approaches are also appended on the extracted sentences. For example, redundancy removal (Carbonell and Goldstein, 1998) and sentence compression (Knight and Marcu, 2000) approaches are used to make the summary more concise. Sentence re-ordering approaches (Barzilay et al., 2002) are used to make the summary more fluent. In most systems, these approaches are treated as independent steps. A sequential process is usually adopted in their implementation, applying the various approaches one after another.

In this paper, we suggest a new summarization framework aiming at integrating multiple objectives of multi-document summarization. The main idea of the approach is to employ a hierarchical summarization process which is motivated by the behavior of a human summarizer. While the document set may be very large in multi-document summarization, the length of the summary to be generated is usually limited. So there are always some concepts that can not be included in the summary. A natural thought is that more general concepts should be considered first. So, when a human summarizer faces a set of many documents, he may follow a general-specific principle to write the summary. The human summarizer may start with finding the core topic in a document set and write some sentences to describe this core topic. Next he may go to find the important sub-topics and cover the subtopics one by one in the summary, then the sub-sub-topics, sub-sub-sub-topics and so on. By this process, the written summary can convey the most salient concepts. Also, the general-specific relation can be used to serve other objectives, i.e. diversity, coherence and etc.

Motivated by this experience, we propose a hierarchical summarization approach which attempts to mimic the behavior of a human summarizer. The approach includes two phases. In the first phase, a hierarchical tree is constructed to organize the important concepts in a document set following the general-to-specific order. In the second phase, an iterative algorithm is proposed to select the sentences based on the constructed hierarchical tree with consideration of the various objectives of multi-document summarization.

2 Word Hierarchical Representation

2.1 Candidate Word Identification

As a matter of fact, the concepts in the original document set are not all necessary to be included in the summary. Therefore, before constructing the hierarchical representation, we first conduct a

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 113–116,
Suntec, Singapore, 4 August 2009.

filtering process to remove the unnecessary concepts in the document set in order to improve the accuracy of the hierarchical representation. In this study, concepts are represented in terms of words. Two types of unnecessary words are considered. One is irrelevant words that are not related to the given query. The other is general words that are not significant for the specified document set. The two types of words are filtered through two features, i.e. *query-relevance* and *topic-specificity*.

The *query-relevance* of a word is defined as the proportion of the number of sentences that contains both the word and at least one query word to the number of sentences that contains the word. If a feature value is large, it means that the co-occurrence rate of the word and the query is high, thus it is more related to the query. The *topic-specificity* of a word is defined as the entropy of its frequencies in different document sets. If the feature value is large, it means that the word appears uniformly in document sets, so its significance to a specified document set is low. Thus, the words with very low query-relevance or with very high topic-specificity are filtered out[1].

2.2 Word Relation Identification and Hierarchical Representation

To construct a hierarchical representation for the words in a given document set, we follow the idea introduced by Lawrie et al. (2001) who use the subsuming relation to express the general-to-specific structure of a document set. A subsumption is defined as an association of two words if one word can be regarded as a sub-concept of the other one. In our approach, the pointwise mutual information (PMI) is used to identify the subsumption between words. Generally, two words with a high PMI is regarded as related. Using the identified relations, the word hierarchical tree is constructed in a top-bottom manner. Two constraints are used in the tree construction process:

(1) For two words related by a subsumption relation, the one which appears more frequently in the document set serves as the parent node in the tree and the other one serves as the child node.

(2) For a word, its parent node in the hierarchical tree is defined as the most related word, which is identified by PMI.

[1] Experimental thresholds are used on the evaluated data.

[2] http://duc.nist.gov/

The construction algorithm is detailed below.

Algorithm 1: Hierarchical Tree Construction

1: Sort the identified key words by their frequency in the document set in descending order, denoted as $T = \{t_1, t_2, \ldots, t_n\}$

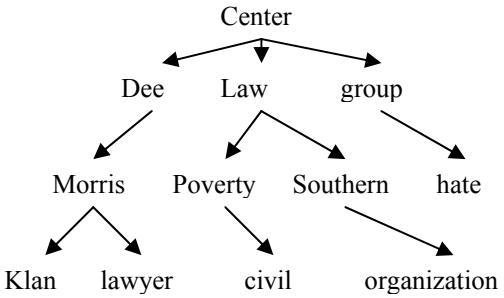
2: For each t_i, i from 1 to n, find the most relevant word t_j from all the words before t_i in T, as $T_i = \{t_1, t_2, \ldots, t_{i-1}\}$. Here the relevance of two words is calculated by their PMI, i.e.

$$PMI(t_i, t_j) = \log \frac{freq(t_i, t_j) * N}{freq(t_i)\, freq(t_j)}$$

If the coverage rate of word t_i by word t_j $P(t_i \mid t_j) = \frac{freq(t_i, t_j)}{freq(t_i)} \geq 0.2$, t_i is regarded as being subsumed by t_j. Here $freq(t_i)$ is the frequency of t_i in the document set and $freq(t_i, t_i)$ is the co-occurrence of t_i and t_j in the same sentences of the document set. N is the total number of tokens in the document set.

4: After all the subsumption relations are found, the tree is constructed by connecting the related words from the first word t_1.

An example of a tree fragment is demonstrated below. The tree is constructed on the document set D0701A from DUC 2007[2], the query of this document set is "Describe the activities of Morris Dees and the Southern Poverty Law Center".



3 Summarization based on Word Hierarchical Representation

3.1 Word Significance Estimation

In order to include the most significant concepts into the summary, before using the hierarchical tree to create an extract, we need to estimate the significance of the words on the tree first. Initially, a rough estimation of the significance of a word is given by its frequency in the document set. However, this simple frequency-based measure is obviously not accurate. One thing we observe from the constructed hierarchical tree is that a word which subsumes many other words is usually very important, though it may not appear

frequently in the document set. The reason is that the word covers many key concepts so it is dominant in the document set. Motivated by this, we develop a bottom-up algorithm which propagates the significance of the child nodes in the hierarchical tree backward to their parent nodes to boost the significance of nodes with many descendants.

Algorithm 2: Word Scoring Theme

1: Set the initial score of each word in T as its log-frequency, i.e. $score(t_i) = \log freq(t_i)$.
2: For t_i from n to 1, propagate an importance score to its parent node $par(t_i)$ (if exists) according to their relevance, i.e. $score(par(t_i)) = score(par(t_i)) + \log freq(t_i, par(t_i))$.

3.2 Sentence Selection

Based on the word hierarchical tree and the estimated word significance, we propose an iterative algorithm to select sentences which is able to integrate the multiple objectives for composing a relevant, concise and fluent summary. The algorithm follows a general-to-specific order to select sentences into the summary. In the implementation, the idea is carried out by following a top-down order to cover the words in the hierarchical tree. In the beginning, we consider several "seed" words which are in the top-level of the tree (these words are regarded as the core concepts in the document set). Once some sentences have been extracted according to these "seed" words, the algorithm moves to down-level words through the subsumption relations between the words. Then new sentences are added according to the down-level words and the algorithm continues moving to lower levels of the tree until the whole summary is generated. For the purpose of reducing redundancy, the words already covered by the extracted sentences will be ignored while selecting new sentences. To improve the fluency of the generated summary, after a sentence is selected, it is inserted to the position according to the subsumption relation between the words of this sentence and the sentences which are already in the summary. The detailed process of the sentence selection algorithm is described below.

Algorithm 3: Summary Generation

1: For the words in the hierarchical tree, set the initial states of the top n words[3] as "activated" and the states of other words as "inactivated".
2: For all the sentences in the document set, select the sentence with the largest score according to the "activated" word set. The score of a sentence s is defined as $score(s) = \frac{1}{|s|}\sum score(t_i)$ where t_i is a word belongs to s and the state of t_i should be "activated". $|s|$ is the number of words in s.
3: For the selected sentence s_k, the subsumption relations between it and the existing sentences in the current summary are calculated and the most related sentence s_l is selected. s_k is then inserted to the position right behind s_l.
4: For each word t_i belongs to the selected sentence s_k, set its state to "inactivated"; for each word t_j which is subsumed by t_i, set its state to "activated".
5: Repeat step 2-4 until the length limit of the summary is exceeded.

4 Experiment

Experiments are conducted on the DUC 2007 data set which contains 45 document sets. Each document set consists of 25 documents and a topic description as the query. In the task definition, the length of the summary is limited to 250 words. In our summarization system, pre-processing includes stop-word removal and word stemming (conducted by GATE[4]).

One of the DUC evaluation methods, ROUGE (Lin and Hovy, 2003), is used to evaluate the content of the generated summaries. ROUGE is a state-of-the-art automatic evaluation method based on *N*-gram matching between system summaries and human summaries. In the experiment, our system is compared to the top systems in DUC 2007. Moreover, a baseline system which considers only the frequencies of words but ignores the relations between words is included for comparison. Table 1 below shows the average recalls of ROUGE-1, ROUGE-2 and ROUGE-SU4 over the 45 DUC 2007document sets. In the experiment, the proposed summarization system outperforms the baseline system, which proves the benefit of considering the relations between words. Also, the system ranks the 6th among the 32 submitted systems in DUC 2007. This shows that the proposed approach is competitive.

	ROUGE-1	ROUGE-2	ROUGE-SU4
S15	0.4451	0.1245	0.1771
S29	0.4325	0.1203	0.1707
S4	0.4342	0.1189	0.1699
S24	0.4526	0.1179	0.1759

[3] n is set to 3 experimentally on the evaluation data set.

[4] http://gate.ac.uk/

S13	0.4218	0.1117	0.1644
Ours	0.4257	0.1110	0.1608
Baseline	0.4088	0.1040	0.1542

Table 1. ROUGE Evaluation Results

To demonstrate the advantage of the proposed approach, i.e. its ability to incorporate multiple summarization objectives, the fragments of the generated summaries on the data set D0701A are also provided below as a case study.

The summary produced by our system
The Southern Poverty Law Center tracks hate groups, and Intelligence Report covers right-wing extremists. Morris Dees, co-founder of the Southern Poverty Law Center in Montgomery, Ala. Dees, founder of the Southern Poverty Law Center, has won a series of civil right suits against the Ku Klux Klan and other racist organizations in a campaign to drive them out of business. In 1987, Dees won a $7 million verdict against a Ku Klux Klan organization over the slaying of a 19-year-old black man in Mobile, Ala.
The summary produced by the baseline system
Morris Dees, co-founder of the Southern Poverty Law Center in Montgomery, Ala. The Southern Poverty Law Center tracks hate groups, and Intelligence Report covers right-wing extremists. The Southern Poverty Law Center previously recorded a 20-percent increase in hate groups from 1996 to 1997. The verdict was obtained by lawyers for the Southern Poverty Law Center, a nonprofit organization in Birmingham, Ala.

Comparing the generated summaries of the two systems, we can see that the summary generated by the proposed approach is better in coherence and fluency since these factors are considered in the integrated summarization framework. Various summarization approaches, i.e. sentence ranking, redundancy removal and sentence re-ordering, are all implemented in the sentence selection algorithm based on the word hierarchical tree. However, we also observe that the proposed approach fails to generate better summaries on some document sets. The main problem is that the quality of the constructed hierarchical tree is not always satisfied. In the proposed summarization approach, we mainly rely on the PMI between the words to construct the hierarchical tree. However, a single PMI-based measure is not enough to characterize the word relation. Consequently the constructed tree can not always well represent the concepts for some document sets. Another problem is that the two constraints used in the tree construction algorithm are not always right in real data. So we regard developing better tree construction approaches as of primary importance. Also, there are other places which can be improved in the future, such as the word significance estimation and sentence inserting algorithms. Nevertheless, we believe that the idea of incorporating the multiple summarization objectives into one integrated framework is meaningful and worth further study.

5 Conclusion

We introduced a summarization framework which aims at integrating various summarization objectives. By constructing a hierarchical tree representation for the words in the original document set, we proposed a summarization approach for the purpose of generating a relevant, concise and fluent summary. Experiments on DUC 2007 showed the advantages of the integrated framework.

Acknowledgments

The work described in this paper was partially supported by Hong Kong RGC Projects (No. PolyU 5217/07E) and partially supported by The Hong Kong Polytechnic University internal grants (A-PA6L and G-YG80).

References

R. Barzilay, N. Elhadad, and K. R. McKeown. 2002. *Inferring strategies for sentence ordering in multidocument news summarization*. Journal of Artificial Intelligence Research, 17:35-55, 2002.

J. Carbonell and J. Goldstein. 1998. *The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries*. In Proceedings of ACM SIGIR 1998, pp 335-336.

K. Knight and D. Marcu. 2000. *Statistics-based summarization --- step one: Sentence compression.* In Proceeding of The American Association for Artificial Intelligence Conference (AAAI-2000), pp 703-710.

D. Lawrie, W. B. Croft and A. Rosenberg. 2001. *Finding topic words for hierarchical summarization*. In Proceedings of ACM SIGIR 2001, pp 349-357.

C. Lin and E. Hovy. 2003. *Automatic evaluation of summaries using n-gram co-occurance statistics*. In Proc. of HLT-NAACL 2003, pp 71-78.

Co-Feedback Ranking for Query-Focused Summarization

Furu Wei[1,2,3] Wenjie Li[1] and Yanxiang He[2]

[1] The Hong Kong Polytechnic University, Hong Kong [2] Wuhan University, China
{csfwei,cswjli}@comp.polyu.edu.hk {frwei,yxhe}@whu.edu.cn

[3] IBM China Research Laboratory, Beijing, China

Abstract

In this paper, we propose a novel ranking framework – Co-Feedback Ranking (Co-FRank), which allows two base rankers to supervise each other during the ranking process by providing their own ranking results as feedback to the other parties so as to boost the ranking performance. The mutual ranking refinement process continues until the two base rankers cannot learn from each other any more. The overall performance is improved by the enhancement of the base rankers through the mutual learning mechanism. We apply this framework to the sentence ranking problem in query-focused summarization and evaluate its effectiveness on the DUC 2005 data set. The results are promising.

1 Introduction and Background

Sentence ranking is the issue of most concern in extractive summarization. Feature-based approaches rank the sentences based on the features elaborately designed to characterize the different aspects of the sentences. They have been extensively investigated in the past due to their easy implementation and the ability to achieve promising results. The use of feature-based ranking has led to many successful (e.g. top five) systems in DUC 2005-2007 query-focused summarization (Over et al., 2007). A variety of statistical and linguistic features, such as term distribution, sentence length, sentence position, and named entity, etc., can be found in literature. Among them, query relevance, centroid (Radev et al., 2004) and signature term (Lin and Hovy, 2000) are most remarkable.

There are two alternative approaches to integrate the features. One is to combine features into a unified representation first, and then use it to rank the sentences. The other is to utilize rank fusion or rank aggregation techniques to combine the ranking results (orders, ranks or scores) produced by the multiple ranking functions into a unified rank. The most popular implementation of the latter approaches is to linearly combine the features to obtain an overall score which is then used as the ranking criterion. The weights of the features are either experimentally tuned or automatically derived by applying learning-based mechanisms. However, both of the above-mentioned "*combine-then-rank*" and "*rank-then-combine*" approaches have a common drawback. They do not make full use of the information provided by the different ranking functions and neglect the interaction among them before combination. We believe that each individual ranking function (we call it base ranker) is able to provide valuable information to the other base rankers such that they learn from each other by means of mutual ranking refinement, which in turn results in overall improvement in ranking. To the best of our knowledge, this is a research area that has not been well addressed in the past.

The inspiration for the work presented in this paper comes from the idea of Co-Training (Blum and Mitchell, 1998), which is a very successful paradigm in the semi-supervised learning framework for classification. In essence, co-training employs two weak classifiers that help augment each other to boost the performance of the learning algorithms. Two classifiers mutually cooperate with each other by providing their own labeling results to enrich the training data for the other parties during the supervised learning process. Analogously, in the context of ranking, although each base ranker cannot decide the overall ranking well on itself, its ranking results indeed reflect its opinion towards the ranking from its point of view. The two base rankers can then share their own opinions by providing the ranking results to each other as feedback. For each ranker, the feedback from the other ranker contains additional information to guide the refinement of its ranking results if the feedback is defined and used appropriately. This process continues until the two base rankers can not learn from each other any more. We call this ranking paradigm Co-Feedback Ranking (Co-FRank). The way how to use the feedback information

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 117–120,
Suntec, Singapore, 4 August 2009.

varies depending on the nature of a ranking task. In this paper, we particularly consider the task of query-focused summarization. We design a new sentence ranking algorithm which allows a query-dependent ranker and a query-independent ranker mutually learn from each other under the Co-FRank framework.

2 Co-Feedback Ranking for Query-Focused Summarization

2.1 Co-Feedback Ranking Framework

Given a set of objects O, one can define two base ranker f_1 and f_2: $f_1(o) \to \Re, f_2(o) \to \Re, \forall o \in O$. The ranking results produced by f_1 and f_2 individually are by no means perfect but the two rankers can provide relatively reasonable ranking information to supervise each other so as to jointly improve themselves. One way to do Co-Feedback ranking is to take the most confident ranking results (e.g. highly ranked instances based on orders, ranks or scores) from one base ranker as feedback to update the other's ranking results, and vice versa. This process continues iteratively until the termination condition is reached, as depicted in Procedure 1. While the standard Co-Training algorithm requires two sufficient and redundant views, we suggest f_1 and f_2 be two independent rankers which emphasize two different aspects of the objects in O.

Procedure 1. *Co-FRank*(f_1, f_2, O)
1: Rank O with f_1 and obtain the ranking results r_1;
2: Rank O with f_2 and obtain the ranking results r_2;
3: **Repeat**
4: Select the top N ranked objects τ_1 from r_1 as feedback to supervise f_2, and re-rank O using f_2 and τ_1; Update r_2;
5: Select the top N ranked objects τ_2 from r_2 as feedback to supervise f_1, and re-rank O using f_1 and τ_2; Update r_1;
5: **Until** I(O).

The termination condition I(O) can be defined according to different application scenarios. For example, I(O) may require the top K ranked objects in r_1 and r_2 to be identical if one is particularly interested in the top ranked objects. It is also very likely that r_1 and r_2 do not change any more after several iterations (or the top K objects do not change). In this case, the two base rankers can not learn from each other any more, and the Co-Feedback ranking process should terminate either. The final ranking results can be easily determined by combining the two base rankers without any parameter, because they have already learnt from each other and can be equally treated.

2.2 Query-Focused Summarization based on Co-FRank

The task of query-focused summarization is to produce a short summary (250 words in length) for a set of related documents D with respect to the query q that reflects a user's information need. We follow the traditional extractive summarization framework in this study, where the two critical processes involved are sentence ranking and sentence selection, yet we focus more on the sentence ranking algorithm based on Co-FRank. As for sentence selection, we incrementally add into the summary the highest ranked sentence if it doesn't significantly repeat[1] the information already included in the summary until the word limitation is reached.

In the context of query-focused summarization, two kinds of features, i.e. query-dependent and query-independent features are necessary and they are supposed to complement each other. We then use these two kinds of features to develop the two base rankers. The query-dependent feature (i.e. the relevance of the sentence s to the query q) is defined as the cosine similarity between s and q.

$$f_1 \Leftrightarrow rel(s,q) = \cos(s,q) = \vec{s} \bullet \vec{q} / \left\|\vec{s}\right\| \cdot \left\|\vec{q}\right\| \quad (1)$$

The words in s and q vectors are weighted by *tf*isf*. Meanwhile, the query-independent feature (i.e. the sentence significance based on word centroid) is defined as

$$f_2 \Leftrightarrow c(s) = \sum_{w \in s} c(w) / \sqrt{|s|} \quad (2)$$

where $c(w)$ is the centroid weight of the word w in s and $c(w) = \sum_{s \in D} \left(tf_w^s \cdot isf_w\right) / N_S^D$. N_S^D is the total number of the sentences in D, tf_w^s is the frequency of w in s, and $isf_w = \log\left(N_s^D / sf_w\right)$ is the inverse sentence frequency (ISF) of w, where sf_w is the sentence frequency of w in D. The sentence ranking algorithm based on Co-FRank is detailed in the following Algorithm 1.

Algorithm 1. *Co-FRank*(f_1, f_2, D, q)
1: Extract sentences $S=\{s_1, \ldots s_m\}$ from D;
2: Rank S with f_1 and obtain the ranking results r_1;
3: Rank S with f_2 and obtain the ranking results r_2;
4: Normalize r_1, $r_1(s_i) = (r_1(s_i) - \min(r_1)) / (\max(r_1) - \min(r_1))$;
5: Normalize r_2, $r_2(s_i) = (r_2(s_i) - \min(r_2)) / (\max(r_2) - \min(r_2))$;
6: **Repeat**

[1] A sentence is discarded if the cosine similarity of it to any sentence already selected into the summary is greater than 0.9.

7: Select the top N ranked sentences at round n τ_1^n from r_1 as feedback for f_2, and re-rank S using f_2 and τ_1^n,

$$\pi_2(s_i) \leftarrow \sum_{k=1}^{n} sim(s_i, \tau_1^k)/n,\ \pi_2 = \frac{\pi_2 - \min(\pi_2)}{\max(\pi_2) - \min(\pi_2)}$$
$$r_2(s_i) \leftarrow \eta \cdot f_2(s_i) + (1-\eta) \cdot \pi_2(s_i) \quad (3)$$

8: Select the top N ranked sentences at round n τ_2^n from r_2 as feedback for f_1, and re-rank S using f_1 and τ_2^n;

$$\pi_1(s_i) \leftarrow \sum_{k=1}^{n} sim(s_i, \tau_2^k)/n,\ \pi_1 = \frac{\pi_1 - \min(\pi_1)}{\max(\pi_1) - \min(\pi_1)}$$
$$r_1(s_i) \leftarrow \eta \cdot f_1(s_i) + (1-\eta) \cdot \pi_1(s_i) \quad (4)$$

9: **Until** the top K sentences in r_1 and r_2 are the same, both r_1 and r_2 do not change any more, or maximum iteration round is achieved.

10: Calculate the final ranking results,

$$r(s_i) = (r_1(s_i) + r_2(s_i))/2. \quad (5)$$

The update strategies used in Algorithm 1, as formulated in Formulas (3) and (4), are designed based on the intuition that the new ranking of the sentence s from one base ranker (say f_1) consists of two parts. The first part is the initial ranking produced by f_1. The second part is the similarity between s and the top N feedback provided by the other ranker (say f_2), and vice versa. The top K ranked sentences by f_2 are supposed to be highly supported by f_2. As a result, a sentence that is similar to those top ranked sentences should deserve a high rank as well. $sim(s_i, \tau_2^n)$ captures the effect of such feedback at round n and the definition of it may vary with regard to the application background. For example, it can be defined as the maximum, the minimum or the average similarity value between s_i and a set of feedback sentences in τ_2. Through this mutual interaction, the two base rankers supervise each other and are expected as a whole to produce more reliable ranking results.

We assume each base ranker is most confident with its first ranked sentence and set N to 1. Accordingly, $sim(s_i, \tau_2^n)$is defined as the similarity between s_i and the one sentence in τ_2^n. η is a balance factor which can be viewed as the proportion of the dependence of the new ranking results on its initial ranking results. K is set to 10 as 10 sentences are basically sufficient for the summarization task we work on. We carry out at most 5 iterations in the current implementation.

3 Experimental Study

We take the DUC 2005 data set as the evaluation corpus in this preliminary study. ROUGE (Lin and Hovy, 2003), which has been officially adopted in the DUC for years is used as the evaluation criterion. For the purpose of comparison, we implement the following two basic ranking functions and the linear combination of them for reference, i.e. the query relevance based ranker (denoted by QRR, same as f_1) and the word centroid based ranker (denoted by WCR, same as f_2), and the linear combined ranker, $LCR = \lambda\ QRR + (1-\lambda)\ WCR$, where λ is a combination parameter. *QRR* and *WCR* are normalized by $(x - \min)/(\max - \min)$, where *x*, *max* and *min* denote the original ranking score, the maximum ranking score and minimum ranking score produced by a ranker, respectively.

Table 1 shows the results of the average recall scores of ROUGE-1, ROUGE-2 and ROUGE-SU4 along with their 95% confidence intervals included within square brackets. Among them, ROUGE-2 is the primary DUC evaluation criterion.

	ROUGE-1	ROUGE-2	ROUGE-SU4
QRR	0.3597 [0.3540, 0.3654]	0.0664 [0.0630, 0.0697]	0.1229 [0.1196, 0.1261]
WCR	0.3504 [0.3436, 0.3565]	0.0644 [0.0614, 0.0675]	0.1171 [0.1138, 0.1202]
LCR*	0.3513 [0.3449, 0.3572]	0.0645 [0.0613, 0.0676]	0.1177 [0.1145, 0.1209]
Co-FRank+	0.3769 [0.3712, 0.3829]	0.0762 [0.0724, 0.0799]	0.1317 [0.1282, 0.1351]
LCR**	0.3753 [0.3692, 0.3813]	0.0757 [0.0719, 0.0796]	0.1302 [0.1265, 0.1340]
Co-FRank++	**0.3783 [0.3719, 0.3852]**	**0.0775 [0.0733, 0.0810]**	**0.1323 [0.1293, 0.1360]**

* The worst results produced by LCR when $\lambda = 0.1$

\+ The worst results produced by Co-FRank when $\eta = 0.6$

** The best results produced by LCR when $\lambda = 0.4$

\++ The best results produced by Co-FRank when $\eta = 0.8$

Table 1 Compare different ranking strategies

Note that the improvement of LCR over QRR and WCR is rather significant if the combination parameter λ is selected appropriately. Besides, Co-FRank is always superior to LCR regardless of the best or the worst ouput, and the improvement is visible. The reason is that both QRR and WCR are enhanced step by step in Co-FRank, which in turn results in the increased overall performance. The trend of the improvement has been clearly observed in the experiments. This observation validates our motivation and the rationality of the algorithm proposed in this paper and motivates our further investigation on this topic.

We continue to examine the parameter settings in LCR and Co-FRank. Table 2 shows the results of LCR when the value of λ changes from 0.1 to

1.0, and Table 3 shows the results of Co-FRank with η ranging from 0.5 to 0.9. Notice that η is not a combination parameter. We believe that a base ranker should have at least half belief in its initial ranking results and thus the value of the η should be greater than 0.5. We find that LCR heavily depends on λ. LCR produces relatively good and stable results with λ varying from 0.4 to 0.6. However, the ROUGE scores drop apparently when λ heading towards its two end values, i.e. 0.1 and 1.0.

λ	ROUGE-1	ROUGE-2	ROUGE-SU4
0.1	0.3513 [0.3449, 0.3572]	0.0645 [0.0613, 0.0676]	0.1177 [0.1145, 0.1209]
0.2	0.3623 [0.3559, 0.3685]	0.0699 [0.0662, 0.0736]	0.1235 [0.1197, 0.1271]
0.3	0.3721 [0.3660, 0.3778]	0.0741 [0.0706, 0.0778]	0.1281 [0.1246, 0.1318]
0.4	0.3753 [0.3692, 0.3813]	0.0757 [0.0719, 0.0796]	0.1302 [0.1265, 0.1340]
0.5	0.3756 [0.3698, 0.3814]	0.0755 [0.0717, 0.0793]	0.1307 [0.1272, 0.1342]
0.6	0.3770 [0.3710, 0.3826]	0.0754 [0.0716, 0.0791]	0.1323 [0.1286, 0.1357]
0.7	0.3698 [0.3636, 0.3759]	0.0718 [0.0680, 0.0756]	0.1284 [0.1246, 0.1318]
0.8	0.3672 [0.3613, 0.3730]	0.0706 [0.0669, 0.0743]	0.1271 [0.1234, 0.1305]
0.9	0.3651 [0.3591, 0.3708]	0.0689 [0.0652, 0.0726]	0.1258 [0.1220, 0.1293]

Table 2 LCR with different λ values

As shown in Table 3, the Co-FRank can always produce stable and promising results regardless of the change of η. More important, even the worst result produced by Co-FRank still outperforms the best result produced by LCR.

η	ROUGE-1	ROUGE-2	ROUGE-SU4
0.5	0.3750 [0.3687, 0.3810]	0.0766 [0.0727, 0.0804]	0.1308 [0.1270, 0.1344]
0.6	0.3769 [0.3712, 0.3829]	0.0762 [0.0724, 0.0799]	0.1317 [0.1282, 0.1351]
0.7	0.3775 [0.3713, 0.3835]	0.0763 [0.0724, 0.0801]	0.1319 [0.1282, 0.1354]
0.8	**0.3783 [0.3719, 0.3852]**	**0.0775 [0.0733, 0.0810]**	**0.1323 [0.1293, 0.1360]**
0.9	0.3779 [0.3722, 0.3835]	0.0765 [0.0728, 0.0803]	0.1319 [0.1285, 0.1354

Table 3 Co-FRank with different η values

We then compare our results to the DUC participating systems. We present the following representative ROUGE results of (1) the top three DUC participating systems according to ROUGE-2 scores (S15, S17 and S10); and (2) the NIST baseline which simply selects the first sentences from the documents.

	ROUGE-1	ROUGE-2	ROUGE-SU4
Co-FRank	**0.3783**	**0.0775**	**0.1323**
S15	-	0.0725	0.1316
S17	-	0.0717	0.1297
S10	-	0.0698	0.1253
Baseline		0.0403	0.0872

Table 4 Compare with DUC participating systems

It is clearly shown in Table 4 that Co-FRank can produce a very competitive result, which significantly outperforms the NIST baseline and meanwhile it is superior to the best participating system in the DUC 2005.

4 Conclusion and Future Work

In this paper, we propose a novel ranking framework, namely Co-Feedback Ranking (Co-FRank), and examine its effectiveness in query-focused summarization. There is still a lot of work to be done on this topic. Although we show the promising achievements of Co-Frank from the perspective of experimental studies, we expect a more theoretical analysis on Co-FRank. Meanwhile, we would like to investigate more appropriate techniques to use feedback, and we are interested in applying Co-FRank to the other applications, such as opinion summarization where the integration of opinion-biased and document-biased ranking is necessary.

Acknowledgments
The work described in this paper was supported by the Hong Kong Polytechnic University internal the grants (G-YG80 and G-YH53) and the China NSF grant (60703008).

References

Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. *In Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pp92-100.

Chin-Yew Lin and Eduard Hovy. 2000. The Automated Acquisition of Topic Signature for Text Summarization. *In Proceedings of COLING*, pp495-501.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. *In Proceedings of HLT-NAACL*, pp71-78.

Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. 2004. Centroid-based Summarization of Multiple Documents. *Information Processing and Management*, 40:919-938.

Paul Over, Hoa Dang and Donna Harman. 2007. DUC in Context. *Information Processing and Management*, 43(6):1506-1520.

Reducing SMT Rule Table with Monolingual Key Phrase

Zhongjun He[†] **Yao Meng**[†] **Yajuan Lü**[‡] **Hao Yu**[†] **Qun Liu**[‡]
[†] Fujitsu R&D Center CO., LTD, Beijing, China
{hezhongjun, mengyao, yu}@cn.fujitsu.com
[‡] Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{lvyajuan, liuqun}@ict.ac.cn

Abstract

This paper presents an effective approach to discard most entries of the rule table for statistical machine translation. The rule table is filtered by monolingual *key phrases*, which are extracted from source text using a technique based on term extraction. Experiments show that 78% of the rule table is reduced without worsening translation performance. In most cases, our approach results in measurable improvements in BLEU score.

1 Introduction

In statistical machine translation (SMT) community, the state-of-the-art method is to use rules that contain hierarchical structures to model translation, such as the hierarchical phrase-based model (Chiang, 2005). Rules are more powerful than conventional phrase pairs because they contain structural information for capturing long distance reorderings. However, hierarchical translation systems often suffer from a large rule table (the collection of rules), which makes decoding slow and memory-consuming.

In the training procedure of SMT systems, numerous rules are extracted from the bilingual corpus. During decoding, however, many of them are rarely used. One of the reasons is that these rules have low quality. The rule quality are usually evaluated by the conditional translation probabilities, which focus on the correspondence between the source and target phrases, while ignore the quality of phrases in a monolingual corpus.

In this paper, we address the problem of reducing the rule table with the information of monolingual corpus. We use $C\text{-}value$, a measurement of automatic term recognition, to score source phrases. A source phrase is regarded as a *key phrase* if its score greater than a threshold. Note that a source phrase is either a *flat* phrase consists of words, or a *hierarchical* phrase consists of both words and variables. For rule table reduction, the rule whose source-side is not key phrase is discarded.

Our approach is different from the previous research. Johnson et al. (2007) reduced the phrase table based on the significance testing of phrase pair co-occurrence in bilingual corpus. The basic difference is that they used statistical information of bilingual corpus while we use that of monolingual corpus. Shen et al. (2008) proposed a string-to-dependency model, which restricted the target-side of a rule by dependency structures. Their approach greatly reduced the rule table, however, caused a slight decrease of translation quality. They obtained improvements by incorporating an additional dependency language model. Different from their research, we restrict rules on the source-side. Furthermore, the system complexity is not increased because no additional model is introduced.

The hierarchical phrase-based model (Chiang, 2005) is used to build a translation system. Experiments show that our approach discards 78% of the rule table without worsening the translation quality.

2 Monolingual Phrase Scoring

2.1 Frequency

The basic metrics for phrase scoring is the frequency that a phrase appears in a monolingual corpus. The more frequent a source phrase appears in a corpus, the greater possibility the rule that contains the source phrase may be used.

However, one limitation of this metrics is that if we filter the rule table by the source phrase with lower frequency, most long phrase pairs will be discarded. Because the longer the phrase is, the less possibility it appears. However, long phrases

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 121–124,
Suntec, Singapore, 4 August 2009.

are very helpful for reducing ambiguity since they contains more information than short phrases.

Another limitation is that the frequency metrics focuses on a phrase appearing by itself while ignores it appears as a substring of longer phrases. It is therefore inadequate for hierarchical phrases.

We use an example for illustration. Considering the following three rules (the subscripts indicate word alignments):

R_1 :
接受$_1$ 布什$_2$ 总统$_3$ 的$_4$ 邀请$_5$
accept$_1$ President$_3$ Bush$_2$'s$_4$ invitation$_5$

R_2 :
接受$_1$ 布什$_2$ X$_3$ 的$_4$ 邀请$_5$
accept$_1$ X$_3$ Bush$_2$'s$_4$ invitation$_5$

R_3 :
接受$_1$ X$_2$ 的$_3$ 邀请$_4$
accept$_1$ X$_2$'s$_3$ invitation$_4$

We use f_1, f_2 and f_3 to represent their source-sides, respectively. The hierarchical phrases f_2 and f_3 are sub-strings of f_1. However, R_3 is suggested to be more useful than R_2. The reason is that f_3 may appears in various phrases, such as "接受 法国 的 邀请, accept France 's invitation". While f_2 almost always appears in f_1, indicating that the variable X may not be replaced with other words expect "President". It indicates that R_2 is not likely to be useful, although f_2 may appears frequently in a corpus.

2.2 C-value

$C\text{-}value$, a measurement of automatic term recognition, is proposed by Frantzi and Ananiadou (1996) to extract nested collocations, collocations that substrings of other longer ones.

We use $C\text{-}value$ for two reasons: on one hand, it uses rich factors besides phrase frequency, e.g. the phrase length, the frequency that a sub-phrase appears in longer phrases. Thus it is appropriate for extracting hierarchical phrases. On the other hand, the computation of $C\text{-}value$ is efficient.

Analogous to (Frantzi and Ananiadou, 1996), we use 4 factors (L, F, S, N) to determine if a phrase p is a key phrase:

1. $L(p)$, the length of p;
2. $F(p)$, the frequency that p appears in a corpus;
3. $S(p)$, the frequency that p appears as a sub-string in other longer phrases;
4. $N(p)$, the number of phrases that contain p as a substring.

Algorithm 1 Key Phrase Extraction

Input: Monolingual Text
Output: Key Phrase Table *KP*

1: Extract candidate phrases
2: **for** all phrases p in length descending order **do**
3: **if** $N(p) = 0$ **then**
4: $C\text{-}value = (L(p) - 1) \times F(p)$
5: **else**
6: $C\text{-}value = (L(p) - 1) \times (F(p) - \frac{S(p)}{N(p)})$
7: **end if**
8: **if** $C\text{-}value \geq \varepsilon$ **then**
9: add p to *KP*
10: **end if**
11: **for** all sub-strings q of p **do**
12: $S(q) = S(q) + F(p) - S(p)$
13: $N(q) = N(q) + 1$
14: **end for**
15: **end for**

Given a monolingual corpus, key phrases can be extracted efficiently according to Algorithm 1.

Firstly (line 1), all possible phrases are extracted as candidates of key phrases. This step is analogous to the rule extraction as described in (Chiang, 2005). The basic difference is that there are no word alignment constraints for monolingual phrase extraction, which therefore results in a substantial number of candidate phrases. We use the following restrictions to limit the phrase number:

1. The length of a candidate phrase is limited to pl;
2. The length of the initial phrase used to create hierarchical phrases is limited to ipl;
3. The number of variables in hierarchical phrases is limited to nv, and there should be at least 1 word between variables;
4. The frequency of a candidate phrase appears in a corpus should be greater than $freq$.

In our experiments, we set $pl = 5$, $ipl = 10$, $nv = 2$, $freq = 3$. Note that the first 3 settings are used in (Chiang, 2005) for rule extraction.

Secondly (line 3 to 7), for each candidate phrase, $C\text{-}value$ is computed according to the phrase appears by itself (line 4) or as a substring of other long phrases (line 6). The $C\text{-}value$ is in direct proportion to the phrase length (L) and occurrences (F, S), while in inverse proportion to the number of phrases that contain the phrase as a substring (N). This overcomes the limitations of frequency measurement. A phrase is regarded as a key phrase if its $C\text{-}value$ is greater than a threshold ε.

Finally (line 11 to 14), $S(q)$ and $N(q)$ are updated for each substring q.

We use the example in Section 2.1 for illustration. The quadruple for f_1 is $(5, 2, 0, 0)$, indicating that the phrase length is 5 and appears 2 times by itself in the corpus. Therefore $C\text{-}value(f_1) = 8$. The quadruple for f_2 is $(4, 2, 2, 1)$, indicating that the phrase length is 4 and appears 2 times in the corpus. However, the occurrences are as a substring of the phrase f_1. Therefore, $C\text{-}value(f_2) = 0$. While the quadruple for f_3 is $(3, 11, 11, 9)$, which indicates that the phrase length is 3 and appears 11 times as a substring in 9 phrases, thus $C\text{-}value(f_3) = 19.6$. Given the threshold $\varepsilon = 5$, f_1 and f_3 are viewed as key phrases. Thus R_2 will be discarded because its source-side is not a key phrase.

3 Experiments

Our experiments were carried out on two language pairs:

- **Chinese-English**: For this task, the corpora are from the NIST evaluation. The parallel corpus [1] consists of 1M sentence pairs . We trained two trigram language models: one on the Xinhua portion of the Gigaword corpus, and the other on the target-side of the parallel corpus. The test sets were NIST MT06 GALE set (06G) and NIST set (06N) and NIST MT08 test set.

- **German-English**: For this task, the corpora are from the WMT [2] evaluation. The parallel corpus contains 1.3M sentence pairs. The target-side was used to train a trigram language model. The test sets were WMT06 and WMT07.

[1] LDC2002E18 (4,000 sentences), LDC2002T01, LDC2003E07, LDC2003E14, LDC2004T07, LDC2005T10, LDC2004T08_HK_Hansards (500,000 sentences)

[2] http://www.statmt.org/wmt07/shared-task.html

For both the tasks, the word alignment were trained by GIZA++ in two translation directions and refined by "grow-diag-final" method (Koehn et al., 2003). The source-side of the parallel corpus is used to extract key phrases.

3.1 Results

We reimplemented the state-of-the-art hierarchical MT system, Hiero (Chiang, 2005), as the baseline system. The results of the experiments are shown in Table 1 and Table 2.

Table 1 shows the $C\text{-}value$ threshold effect on the size of the rule table, as well as the BLEU scores. Originally, 103M and 195M rules are respectively extracted for Chinese-English and German-English. For both the two tasks, about 78% reduction of the rule table (for Chinese-English $\varepsilon = 200$ and for German-English $\varepsilon = 100$) does not worsen translation performance. We achieved improvements in BLEU on most of the test corpora, except a slight decrease (0.06 point) on WMT07.

We also compared the effects of *frequency* and $C\text{-}value$ metrics for the rule table reduction on Chinese-English test sets. The rule table is reduced to the same size (22% of original table) using the two metrics, separately. However, as shown in Table 2, the *frequency* method decreases the BLEU scores, while the $C\text{-}value$ achieves improvements. It indicates that $C\text{-}value$ is more appropriate than *frequency* to evaluate the importance of phrases, because it considers more factors.

With the rule table filtered by key phrases on the source side, the number of source phrases reduces. Therefore during decoding, a source sentence is suggested to be decomposed into a number of "key phrases", which are more reliable than the discarded phrases. Thus the translation quality does not become worse.

3.2 Adding C-value as a Feature

Conventional phrase-based approaches performed phrase segmentation for a source sentence with a uniform distribution. However, they do not consider the weights of source phrases. Although any strings can be phrases, it is believed that some strings are more likely phrases than others. We use $C\text{-}value$ to describe the weight of a phrase in a monolingual corpus and add it as a feature to the translation model:

C-value	Chinese-English				Germany-English		
Threshold ε	Rule Table (%)	06G	06N	08	Rule Table (%)	06	07
0	100%	12.43	28.58	21.57	100%	27.30	27.95
5	61%	12.22	28.40	21.33	54%	27.39	**28.05**
20	44%	12.24	28.29	21.21	37%	27.47	27.94
100	28%	12.36	28.56	21.67	22%	**27.54**	27.89
200	22%	**12.66**	**28.69**	**22.12**	17%	27.26	27.80
300	20%	12.41	27.76	21.52	15%	27.41	27.69
400	18%	11.88	26.98	20.70	13%	27.36	27.76
500	16%	11.65	26.40	20.32	12%	27.25	27.76

Table 1: C-value threshold effect on the rule table size and BLEU scores.

System	Rule Table (%)	06G	06N	08
Baseline	100%	12.43	28.58	21.57
Frequency	22%	12.24	27.77	21.20
C-value	22%	12.66	28.69	22.12^{*}
+CV-Feature	22%	12.89^{*}	29.22^{*+}	22.56^{*+}

Table 2: BLEU scores on the test sets of the Chinese-English task. * means significantly better than baseline at $p < 0.01$. $^{+}$ means significantly better than C-value at $p < 0.05$.

$$h(F_1^J) = \sum_{k=1}^{K} log(C\text{-}value(\widetilde{f}_k)) \qquad (1)$$

where, $\widetilde{f}_k$ is the source-side of a rule.

The results are shown in the row *+CV-Feature* in Table 2. Measurable improvements are obtained on all test corpora of the Chinese-English task by adding the $C\text{-}value$ feature. The improvements over the baseline are statistically significant at $p < 0.01$ by using the significant test method described in (Koehn, 2004).

4 Conclusion

In this paper, we successfully discarded most entries of the rule table with monolingual key phrases. Experiments show that about 78% of the rule table is reduced and the translation quality does not become worse. We achieve measurable improvements by incorporating $C\text{-}value$ into the translation model.

The use of key phrases is one of the simplest method for the rule table reduction. In the future, we will use sophisticated metrics to score phrases and reduce the rule table size with the information of both the source and target sides.

References

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270.

Katerina T. Frantzi and Sophia Ananiadou. 1996. Extracting nested collocations. In *COLING1996*, pages 41–46.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 967–975, Prague, Czech Republic, June.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL 2003*, pages 127–133.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June.

A Statistical Machine Translation Model Based on a Synthetic Synchronous Grammar

Hongfei Jiang, Muyun Yang, Tiejun Zhao, Sheng Li and Bo Wang
School of Computer Science and Technology
Harbin Institute of Technology
{hfjiang,ymy,tjzhao,lisheng,bowang}@mtlab.hit.edu.cn

Abstract

Recently, various synchronous grammars are proposed for syntax-based machine translation, e.g. synchronous context-free grammar and synchronous tree (sequence) substitution grammar, either purely formal or linguistically motivated. Aiming at combining the strengths of different grammars, we describes a synthetic synchronous grammar (SSG), which tentatively in this paper, integrates a synchronous context-free grammar (SCFG) and a synchronous tree sequence substitution grammar (STSSG) for statistical machine translation. The experimental results on NIST MT05 Chinese-to-English test set show that the SSG based translation system achieves significant improvement over three baseline systems.

1 Introduction

The use of various synchronous grammar based formalisms has been a trend for statistical machine translation (SMT) (Wu, 1997; Eisner, 2003; Galley et al., 2006; Chiang, 2007; Zhang et al., 2008). The grammar formalism determines the intrinsic capacities and computational efficiency of the SMT systems.

To evaluate the capacity of a grammar formalism, two factors, i.e. generative power and expressive power are usually considered (Su and Chang, 1990). The generative power refers to the ability to generate the strings of the language, and the expressive power to the ability to describe the same language with fewer or no extra ambiguities. For the current synchronous grammars based SMT, to some extent, the *generalization ability* of the grammar rules (the usability of the rules for the new sentences) can be considered as a kind of the generative power of the grammar and the *disambiguition ability* to the rule candidates can be considered as an embodiment of expressive power.

However, the *generalization ability* and the *disambiguition ability* often contradict each other in practice such that various grammar formalisms in SMT are actually different trade-off between them. For instance, in our investigations for SMT (Section 3.1), the **F**ormally **SCFG** based hierarchical phrase-based model (hereinafter FSCFG) (Chiang, 2007) has a better generalization capability than a **L**inguistically motivated **STSSG** based model (hereinafter LSTSSG) (Zhang et al., 2008), with 5% rules of the former matched by NIST05 test set while only 3.5% rules of the latter matched by the same test set. However, from expressiveness point of view, the former usually results in more ambiguities than the latter.

To combine the strengths of different synchronous grammars, this paper proposes a statistical machine translation model based on a synthetic synchronous grammar (SSG) which syncretizes FSCFG and LSTSSG. Moreover, it is noteworthy that, from the combination point of view, our proposed scheme can be considered as a novel system combination method which goes beyond the existing post-decoding style combination of N-best hypotheses from different systems.

2 The Translation Model Based on the Synthetic Synchronous Grammar

2.1 The Synthetic Synchronous Grammar

Formally, the proposed Synthetic Synchronous Grammar (SSG) is a tuple

$$G = \langle \Sigma_s, \Sigma_t, N_s, N_t, X, \boldsymbol{P} \rangle$$

where $\Sigma_s(\Sigma_t)$ is the alphabet set of source (target) terminals, namely the vocabulary; $N_s(N_t)$ is the alphabet set of source (target) non-terminals, such

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 125–128,
Suntec, Singapore, 4 August 2009.

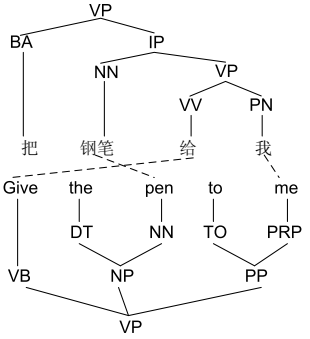


Figure 1: A syntax tree pair example. Dotted lines stands for the word alignments.

as the POS tags and the syntax labels; X represents the special nonterminal label in FSCFG; and P is the grammar rule set which is the core part of a grammar. Every rule r in P is as:

$$r = \langle \alpha, \gamma, A_{NT}, A_T, \bar{\omega} \rangle$$

where $\alpha \in [\{X\}, N_s, \Sigma_s]^+$ is a sequence of one or more source words in Σ_s and nonterminals symbols in $[\{X\}, N_s]$;$\gamma \in [\{X\}, N_t, \Sigma_t]^+$ is a sequence of one or more target words in Σ_t and nonterminals symbols in $[\{X\}, N_t]$; A_T is a many-to-many corresponding set which includes the alignments between the terminal leaf nodes from source and target side, and A_{NT} is a one-to-one corresponding set which includes the synchronizing relations between the non-terminal leaf nodes from source and target side; $\bar{\omega}$ contains feature values associated with each rule.

Through this formalization, we can see that FSCFG rules and LSTSSG rules are both included. However, we should point out that the rules with mixture of X non-terminals and syntactic non-terminals are not included in our current implementation despite that they are legal under the proposed formalism. The rule extraction in current implementation can be considered as a combination of the ones in (Chiang, 2007) and (Zhang et al., 2008). Given the sentence pair in Figure 1, some SSG rules can be extracted as illustrated in Figure 2.

2.2 The SSG-based Translation Model

The translation in our SSG-based translation model can be treated as a SSG derivation. A derivation consists of a sequence of grammar rule applications. To model the derivations as a latent variable, we define the conditional probability distribution over the target translation e and the corresponding derivation d of a given source sentence f as

Input: A source parse tree $T(f_1^J)$
Output: A target translation $\hat{e}$
for $u := 0$ *to* $J-1$ **do**
  for $v := 1$ *to* $J-u$ **do**
    foreach *rule* $r = \langle \alpha, \gamma, A_{NT}, A_T, \bar{\omega} \rangle$ *spanning* $[v, v+u]$ **do**
      if A_{NT} *of* r *is empty* **then**
        Add r into $H[v, v+u]$;
      end
      else
        Substitute the non-terminal leaf node pair (N_{src}, N_{tgt}) with the hypotheses in the hypotheses stack corresponding with N_{src}'s span iteratively.
      end
    end
  end
end
Output the 1-best hypothesis in $H[1, J]$ as the final translation.

Figure 3: The pseudocode for the decoding.

$$(1) \qquad p_\Lambda(\mathbf{d}, \mathbf{e}|\mathbf{f}) = \frac{\exp \sum_k \lambda_k H_k(\mathbf{d}, \mathbf{e}, \mathbf{f})}{\Omega_\Lambda(\mathbf{f})}$$

where H_k is a feature function ,λ_k is the corresponding feature weight and $\Omega_\Lambda(\mathbf{f})$ is a normalization factor for each derivation of $\mathbf{f}$. The main challenge of SSG-based model is how to distinguish and weight the different kinds of derivations . For a simple illustration, using the rules listed in Figure 2, three derivations can be produced for the sentence pair in Figure 1 by the proposed model:

$$d_1 = (R_4, R_1, R_2)$$
$$d_2 = (R_6, R_7, R_8)$$
$$d_3 = (R_4, R_7, R_2)$$

All of them are SSG derivations while d_1 is also a FSCFG derivation, d_2 is also a LSTSSG derivation. Ideally, the model is supposed to be able to weight them differently and to prefer the better derivation, which deserves intensive study. Some sophisticated features can be designed for this issue. For example, some features related with structure richness and grammar consistency[1] of a derivation should be designed to distinguish the derivations involved various heterogeneous rule applications. For the page limit and the fair comparison, we only adopt the conventional features as in (Zhang et al., 2008) in our current implementation.

[1]This relates with reviewers' questions: "can a rule expecting an NN accept an X?" and "...the interaction between the two typed of rules ...". In our study in progress, we would design some features to distinguish the derivation steps which fulfill the expectation or not, to measure how much heterogeneous rules are applied in a derivation and so on.

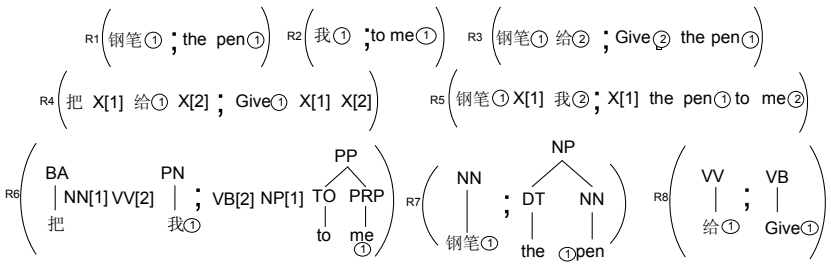


Figure 2: Some synthetic synchronous grammar rules can be extracted from the sentence pair in Figure 1. R_1-R_3 are bilingual phrase rules, R_4-R_5 are FSCFG rules and R_6-R_8 are LSTSSG rules.

2.3 Decoding

For efficiency, our model approximately search for the single 'best' derivation using beam search as

$$(\hat{\mathbf{e}}, \hat{\mathbf{d}}) = \underset{\mathbf{e},\mathbf{d}}{\operatorname{argmax}} \Big\{ \sum_{k} \lambda_k h_k(\mathbf{d}, \mathbf{e}, \mathbf{f}) \Big\}. \quad (2)$$

The major challenge for such a SSG-based decoder is how to apply the heterogeneous rules in a derivation. For example, (Chiang, 2007) adopts a CKY style span-based decoding while (Liu et al., 2006) applies a linguistically syntax node based bottom-up decoding, which are difficult to integrate. Fortunately, our current SSG syncretizes FSCFG and LSTSSG. And the conventional decodings of both FSCFG and LSTSSG are span-based expansion. Thus, it would be a natural way for our SSG-based decoder to conduct a span-based beam search. The search procedure is given by the pseudocode in Figure 3. A hypotheses stack $H[i, j]$ (similar to the "chart cell" in CKY parsing) is arranged for each span $[i, j]$ for storing the translation hypotheses. The hypotheses stacks are ordered such that every span is translated after its possible antecedents: smaller spans before larger spans. For translating each span $[i, j]$, the decoder traverses each usable rule $r = \langle \alpha, \gamma, A_{NT}, A_T, \bar{\omega} \rangle$. If there is no nonterminal leaf node in r, the target side γ will be added into $H[i, j]$ as the candidate hypothesis. Otherwise, the nonterminal leaf nodes in r should be substituted iteratively by the corresponding hypotheses until all nonterminal leaf nodes are processed. The key feature of our decoder is that the derivations are based on synthetic grammar, so that one derivation may consist of applications of heterogeneous rules (Please see d_3 in Section 2.2 as a simple demonstration).

3 Experiments and Discussions

Our system, named HITREE, is implemented in standard C++ and STL. In this section we report on experiments with Chinese-to-English translation base on it. We used FBIS Chinese-to-English parallel corpora (7.2M+9.2M words) as the training data. We also used SRI Language Modeling Toolkit to train a 4-gram language model on the Xinhua portion of the English Gigaword corpus(181M words). NIST MT2002 test set is used as the development set. The NIST MT2005 test set is used as the test set. The evaluation metric is case-sensitive BLEU4. For significant test, we used Zhang's implementation (Zhang et al., 2004)(confidence level of 95%). For comparisons, we used the following three baseline systems:

LSTSSG An in-house implementation of linguistically motivated STSSG based model similar to (Zhang et al., 2008).

FSCFG An in-house implementation of purely formally SCFG based model similar to (Chiang, 2007).

MBR We use an in-house combination system which is an implementation of a classic sentence level combination method based on the Minimum Bayes Risk (MBR) decoding (Kumar and Byrne, 2004).

	Extracted(k)	**S**cored(k)(**S**/**E**%)	**F**iltered(k)(**F**/**S**%)
BP	11,137	4,613(41.4%)	323(0.5%)
LSTSSG	45,580	28,497(62.5%)	984(3.5%)
FSCFG	59,339	25,520(43.0%)	1,266(5.0%)
HITREE	93,782	49,404(52.7%)	1,927(3.9%)

Table 1: The statistics of the counts of the rules in different phases. 'k' means one thousand.

3.1 Statistics of Rule Numbers in Different Phases

Table 1 summarizes the statistics of the rules for different models in three phases: after extraction (*Extracted*), after scoring(*Scored*), and after filtering (*Filtered*) (filtered by NIST05 test set just, similar to the filtering step in phrase-based SMT system). In *Extracted* phase, FSCFG

ID	System	BLEU4	#of used rules(k)
1	LSTSSG	0.2659±0.0043	984
2	FSCFG	0.2613±0.0045	1,266
3	HITREE	0.2730±0.0045	1,927
4	MBR(1,2)	0.2685±0.0044	–

Table 2: The Comparison of LSTSSG, FSCFG ,HITREE and the MBR.

has obvious more rules than LSTSSG. However, in *Scored* phase, this situation reverses. Interestingly, the situation reverses again in *Filtered* phase. The reasons for these phenomenons are that FSCFG abstract rules involves high-degree generalization. Each FSCFG abstract rule averagely have several duplicates[2] in the extracted rule set. Then, the duplicates will be discarded during scoring. However, due to the high-degree generalization , the FSCFG abstract rules are more likely to be matched by the test sentences. Contrastively, LSTSSG rules have more diversified structures and thus weaker generalization capability than FSCFG rules. From the ratios of two transition states, Table 1 indicates that HITREE can be considered as compromise of FSCFG between LSTSSG.

3.2 Overall Performances

The performance comparison results are presented in Table 2. The experimental results show that the SSG-based model (HITREE) achieves significant improvements over the models based on the two isolated grammars: FSCFG and LSTSSG (both $p < 0.001$). From combination point of view, the newly proposed model can be considered as a novel method going beyond the conventional post-decoding style combination methods. The baseline Minimum Bayes Risk combination of LSTSSG based model and FSCFG based model ($MBR(1,2)$) obtains significant improvements over both candidate models (both $p < 0.001$). Meanwhile, the experimental results show that the proposed model outperforms $MBR(1,2)$ significantly ($p < 0.001$). These preliminary results indicate that the proposed SSG-based model is rather promising and it may serve as an alternative, if not superior, to current combination methods.

4 Conclusions

To combine the strengths of different grammars, this paper proposes a statistical machine translation model based on a synthetic synchronous grammar (SSG) which syncretizes a purely formal synchronous context-free grammar (FSCFG) and a linguistically motivated synchronous tree sequence substitution grammar (LSTSSG). Experimental results show that SSG-based model achieves significant improvements over the FSCFG-based model and LSTSSG-based model.

In the future work, we would like to verify the effectiveness of the proposed model on various datasets and to design more sophisticated features. Furthermore, the integrations of more different kinds of synchronous grammars for statistical machine translation will be investigated.

[2]Rules with identical source side and target side are duplicated.

Acknowledgments

This work is supported by the Key Program of National Natural Science Foundation of China (60736014), and the Key Project of the National High Technology Research and Development Program of China (2006AA010108).

References

David Chiang. 2007. Hierarchical phrase-based translation. In *computational linguistics*, 33(2).

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of ACL 2003*.

Galley, M. and Graehl, J. and Knight, K. and Marcu, D. and DeNeefe, S. and Wang, W. and Thayer, I. 2006. Scalable inference and training of context-rich syntactic translation models In *Proceedings of ACL-COLING*.

S. Kumar and W. Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *HLT-04*.

Yang Liu, Qun Liu, Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of ACL-COLING*.

Keh-Yin Su and Jing-Shin Chang. 1990. Some key Issues in Designing Machine Translation Systems. *Machine Translation*, 5(4):265-300.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377-403.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of LREC 2004*, pages 2051-2054.

Min Zhang, Hongfei Jiang, Ai Ti AW, Haizhou Li, Chew Lim Tan and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-HLT*.

English-Chinese Bi-Directional OOV Translation based on Web Mining and Supervised Learning

Yuejie Zhang, Yang Wang and Xiangyang Xue
School of Computer Science
Shanghai Key Laboratory of Intelligent Information Processing
Fudan University, Shanghai 200433, P.R. China
{yjzhang,072021176,xyxue}@fudan.edu.cn

Abstract

In Cross-Language Information Retrieval (CLIR), Out-of-Vocabulary (OOV) detection and translation pair relevance evaluation still remain as key problems. In this paper, an English-Chinese Bi-Directional OOV translation model is presented, which utilizes Web mining as the corpus source to collect translation pairs and combines supervised learning to evaluate their association degree. The experimental results show that the proposed model can successfully filter the most possible translation candidate with the lower computational cost, and improve the OOV translation ranking effect, especially for popular new words.

1 Introduction

In Cross-Language Information Retrieval (CLIR), most of queries are generally composed of short terms, in which there are many Out-of-Vocabulary (OOV) terms like named entities, new words, terminologies and so on. The translation quality of OOVs directly influences the precision of querying relevant multilingual information. Therefore, OOV translation has become a very important and challenging issue in CLIR.

The translation of OOVs can either be acquired from parallel or comparable corpus (Lee, 2006) or mining from Web (Lu, 2004). However, how to evaluate the degree of association between source query term and its target translation is quite important. In this paper, an OOV translation model is established based on the combination pattern of Web mining and translation ranking. Given an OOV, its related information are gotten from search results by search engine, from which the possible translation terms in target language can be extracted and then ranked through supervised learning such as Support Vector Machine (SVM) and Ranking-SVM (Cao, 2006). The basic framework of the translation model is shown in Figure 1.

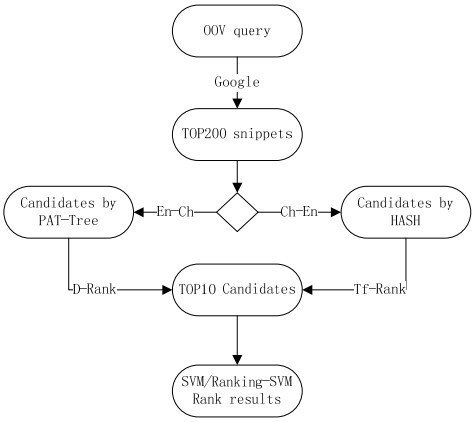


Figure 1. The basic framework of English-Chinese Bi-Directional OOV translation model.

2 Related Research Work

With the rapid growth of Web information, increasing new terms and terminologies cannot be found in bilingual dictionaries. The state-of-art OOV translation strategies tend to use Web itself as a big corpus (Wang, 2004; Zhang, 2004). The quick and direct way of getting required information from Web pages is to use search engines, such as Google, Altavista or Yahoo. Therefore, many OOV translation models based on Web mining are proposed by researchers (Fang, 2006; Wu, 2007).

By introducing supervised learning mechanism, the relevance between original OOV term and extracted candidate translation can be accurately evaluated. Meanwhile, the model proposed exhibits better applicability and can also be applied in processing OOVs with different classes.

3 Chinese OOV Extraction based on PAT-Tree

For a language that has no words boundary like Chinese, PAT-Tree data structure is adopted to extract OOV terms (Chien, 1997). The most outstanding property of this structure is its Semi Infinite String, which can store all the semi-strings of whole corpus in a binary tree. In this tree, branch nodes indicate direction of search

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 129–132,
Suntec, Singapore, 4 August 2009.

and child nodes store information about index and frequency of semi infinite strings. With common strings being extracted, large amounts of noisy terms and fragments are also extracted. For example, when searching for the translation of English abbreviation term "*FDA*", some noisy Chinese terms are extracted, such as "国食品" (17 times), "美国食品" (16 times), "美国食品药" (9 times). In order to filter noisy fragments, the simplified Local-Maxima algorithm is used (Wang, 2004).

4 Translation Ranking based on Supervised Learning

4.1 Ranking by Classification and Ordinal Regression

Based on the extracted terms, the correct translation can be chosen further. A direct option is to rank them by their frequency or length. It works well when the OOV term has a unique meaning and all the Web snippets are about the same topic. However, in much more cases only the highly related fragments of OOV terms can be found, rather than their correct translations. To evaluate the relevance of translation pair precisely, SVM and Ranking-SVM are employed as classifier and ordinal regression model respectively.

4.2 Feature Representation

The same feature set is utilized by SVM and Ranking-SVM.

(1) *Term frequency*: f_q denotes the frequency of OOV to be translated in all the Web snippets of search results. tf_i indicates the number of the translation candidate in all the snippets. df_i represents the number of Web snippets that contains the candidate. df_t means the number of snippets that contains both OOV to be translated and the candidate.

(2) *Term length*: *Len*( ) is the length of the candidate.

(3) *Cooccurrence Distance*: *C-Dist* is the average distance between the OOV query and the translation candidate, computed as follows.

$$C\text{-}Dist = \frac{Sum(Dist)}{df_t} \quad (1)$$

where *Sum*(*Dist*) is the sum of distance in each translation pair of every snippet.

(4) *Length Ratio*: This is the ratio of OOV query length and translation candidate length.

(5) *Rank Value*:

i. *Top Rank* (*T-Rank*): The rank of snippet that first contains the candidate. This value indicates the rank given by search engine.

ii. *Average_Rank* (*A-Rank*): It is the average position of candidate in snippets of search results, shown as follows.

$$A - Rank = \frac{Sum(Rank)}{df_i} \quad (2)$$

where *Sum*(*Rank*) denotes the sum of every single rank value of snippets that contains the candidate.

iii. *Simple_Rank* (*S-Rank*): It is computed based on $Rank(i)=tf_i*Len(i)$, which aims at investigating the impact of these two features on ranking translation.

iv. *R-Rank*: This rank method is utilized as a comparison basis, computed as follows.

$$R - Rank = \alpha \times \frac{|S_n|}{L} + (1-\alpha) \times \frac{f_n}{f_{OOV}} \quad (3)$$

where α is set as 0.25 empirically, $|S_n|$ represents the length of candidate term, L is the largest length of candidate terms, f_n is tf_i, and f_{oov} is f_q in Feature (1).

v. *Df_Rank* (*D-Rank*): It is similar to *S-Rank* and computed based on $Rank(i)= df_i *Len(i)$.

(6) *Mark feature*: Within a certain distance (usually less than 10 characters) between the original OOV and candidate, if there is such a term like "全称", "中文叫", "中文译为", "中文名称", "中文称为", "或称为", "又称为", "英文叫", "英文名为", this feature will be labeled as "+1", else "-1" instead.

Among these features above, some features come from search engine like (1) and (5) and some ones from heuristic rules like (3) and (6). Through the establishment of feature set, the translation candidate can be optimized efficiently and the noisy information can also be filtered.

5 Experiment and Analysis

5.1 Data Set

For the performance evaluation of Chinese-English OOV translation, the corpus of NER task in SIGHAN 2008 provided by Peking University is used. The whole corpus contains 19,866 person names, 22,212 location names and 7,837 organization names, from which 100 person names, 100 location names and 100 organization names are selected for testing. Meanwhile, 300 English named entities are chosen randomly from the terms of 9 categories, which include movie name, book title, organization name, brand name, terminology, idiom, rare animal name, person name

and so on. These new terms are used as the testing data for English-Chinese OOV translation.

5.2 Evaluation Metrics

Three parameters are used for the evaluation of translation and ranking candidates.

$$N-Inclusion-Rate = \frac{number\ of\ correct\ translation\ in\ top\ N\ translations}{total\ number\ of\ OOV\ terms\ to\ be\ translated} \quad (4)$$

$$R-Precision(term_i) = \frac{number\ of\ correct\ transaltion\ in\ top\ R\ translations}{number\ of\ correct\ translations\ for\ term_i\ to\ be\ translated} \quad (5)$$

$$R-Precision = \frac{\sum_{i=1}^{T} R-Precision(term_i)}{total\ number\ of\ OOV\ terms\ to\ be\ translated} \quad (6)$$

where T denotes the number of testing entities. The first one is a measurement for translation and the others are used for ranking measurement.

5.3 Experiment on Parameter Setting

Frequency and length are two crucial features for translation candidates. To get the most related terms into top 10 before the final ranking, a pre-rank testing is performed based on *S-Rank*, *R-Rank* and *D-Rank*. It can be seen from Figure 2 that the pre-rank by *D-Rank* exhibits better performance in translation experiment.

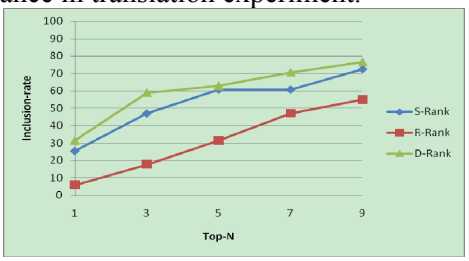


Figure 2. The impact of different *Pre-Rank* manners on English-Chinese OOV translation.

In search results, for some English OOV terms such as "*BYOB*(自带酒水)", there are few candidates with better quality in top 20 snippets. Therefore, in order to find how many snippets are suitable in translation, the experiment on snippet number is performed. It can be observed from Figure 3 that the best performance can be obtained by utilizing 200 snippets.

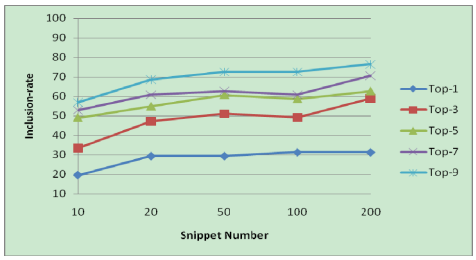


Figure 3. The impact of different snippet number on English-Chinese OOV translation.

5.4 Experiment On English-Chinese Bi-Directional OOV Translation

The experimental results on 300 English new terms are shown in Table 1.

N-Inclusion-Rate	English-Chinese OOV Translation
Top-1	0.313
Top-3	0.587
Top-5	0.627
Top-7	0.707
Top-9	0.763

Table 1. The experimental results on English-Chinese OOV translation.

The experimental results on 300 Chinese named entities are shown in Table 2.

N-Inclusion-Rate	Person Name	Location Name	Organization Name
Top-1	0.210	0.510	0.110
Top-3	0.390	0.800	0.280
Top-5	0.490	0.900	0.400
Top-7	0.530	0.920	0.480
Top-9	0.540	0.930	0.630

Table 2. The experimental results on Chinese-English OOV translation.

It can be observed from Table 2 that the performance of Chinese location name translation is much higher than the other two categories. This is because most of the location names are famous cities or countries. The experimental results above demonstrate that the proposed model can be applicable in all kinds of OOV terms.

5.5 Experiment on Ranking

In SVM-based and Ranking-SVM-based ranking experiment, the statistics on training data are shown in Table 3. For SVM training data, the "*Related*" candidates are neglected. The experimental results on ranking in English-Chinese and Chinese-English OOV translation are shown in Table 4 and 5 respectively.

Number of Candidates	*Correct*	*Related*	*Indifferent*
English-Chinese	234	141	250
Chinese-English	240	144	373

Table 3. Statistics of training data for ranking.

English-Chinese	Top-1 *Inclusion*	Top-3 *Inclusion*	*R-Precision*
D-Rank	0.313	0.587	0.417
T-Rank	0.217	0.430	0.217
SVM	0.530	0.687	0.533
Ranking-SVM	**0.550**	**0.687**	**0.547**

Table 4. The experimental results on ranking in English-Chinese OOV translation.

Chinese-English	Top-1 *Inclusion*	Top-3 *Inclusion*	*R-Precision*
TF-Rank	0.277	0.490	0.287
T-Rank	0.197	0.387	0.207
SVM	0.347	0.587	0.347
Ranking-SVM	**0.357**	**0.613**	**0.387**

Table 5. The experimental results on ranking in Chinese-English OOV translation.

From the experiments above, it can be concluded that the supervised learning significantly outperform the conventional ranking strategies.

5.6 Analysis and Discussion

Through analysis about the experimental results in extraction and ranking, it can be observed that the OOV translation quality is highly related to the following aspects.

(1) The translation results are related to the search engine used, especially for some specific OOV terms. For example, given a query OOV term "两岸三通", the mining result based on Google in China is "*three direct links*", while some meaningless information is mined by the other engines like Live Trans.

(2) Some terms are conventional terminologies and cannot be translated literally. For example, "*woman pace-setter*", a proper name with the particular Chinese characteristic, should be translated into "三八红旗手", rather than "女子的步伐" or "制定".

(3) The proposed model is sensitive to the notability degree of OOV term. For famous person name and book title, the translation performance is very promising. However, for other OOV terms with lower notability, such as "贝尔曼来" and "兰红光", the correct translation cannot even be retrieved by search engine.

(4) Word Sense Disambiguation (WSD) should be added to improve the whole translation performance. Although most of OOVs have unique semantic definition, there are still a few OOVs with ambiguity. For example, "*Rice*" can either be a person name or a kind of food. Another example is "*AARP*", which also has two kinds of meaning, that is, "美国退休者协会" and "地址解析协议".

6 Conclusions and Future Work

In this paper, the proposed model improves the acquirement ability for OOV translation through Web mining and solves the translation pair evaluation problem in a novel way by introducing supervised learning in translation ranking. In addition, it is very significant to apply the key techniques in traditional machine translation into OOV translation, such as OOV recognition, statistical machine learning, alignment of sentence and phoneme, and WSD. The merits of these techniques should be integrated. All these aspects above will become the research focus in our future work.

Acknowledgments

This paper is supported by National Natural Science Foundation of China (No. 60773124), National Science and Technology Pillar Program of China (No. 2007BAH09B03) and Shanghai Municipal R&D Foundation (No. 08dz1500109). Yang Wang is the corresponding author.

References

Chun-Jen Lee, Jason S. Chang, and Jyh-Shing R. Jang. 2006. *Alignment of Bilingual Named Entities in Parallel Corpora Using Statistical Models and Multiple Knowledge Sources.* ACM Transactions on Asian Language Processing, 5(2):121-145.

Gaolin Fang, Hao Yu, and Fumihito Nishino. 2006. *Chinese-English Term Translation Mining Based on Semantic Prediction.* In Proceedings of the COLING/ACL on Main Conference Poster Sessions, pp.199-206.

Jenq-Haur Wang, Jei-Wen Teng, Pu-Jen Cheng, Wen-Hsiang Lu, and Lee-Feng Chien. 2004. *Translating Unknown Cross-Lingual Queries in Digital Libraries Using a Web-based Approach.* In Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries, pp.108-116.

Jian-Cheng Wu and Jason S. Chang. 2007. *Learning to Find English to Chinese Transliterations on the Web.* In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp.996-1004.

L. F. Chien. 1997. *PAT-Tree-Based Keyword Extraction for Chinese Information Retrieval.* In Proceedings of SIGIR'97, pp.50-58.

Wen-Hsiang Lu and Lee-Feng Chien. 2004. *Anchor Text Mining for Translation of Web Queries*: *A Transitive Translation Approach.* ACM Transactions on Information Systems, 22(2): 242-269.

Ying Zhang and Phil Vines. 2004. *Detection and Translation of OOV Terms Prior to Query Time.* In Proceedings of SIGIR'04, pp.524-525.

Yunbo Cao, Jun Xu, Tie-Yan LIU, Hang Li, Yalou HUANG, and Hsiao-Wuen HON. 2006. *Adapting Ranking SVM to Document Retrieval.* In Proceedings of SIGIR'06, pp.186-193.

The Back-translation Score: Automatic MT Evaluation at the Sentence Level without Reference Translations

Reinhard Rapp
Universitat Rovira i Virgili
Avinguda Catalunya, 35
43002 Tarragona, Spain
reinhard.rapp@urv.cat

Abstract

Automatic tools for machine translation (MT) evaluation such as BLEU are well established, but have the drawbacks that they do not perform well at the sentence level and that they presuppose manually translated reference texts. Assuming that the MT system to be evaluated can deal with both directions of a language pair, in this research we suggest to conduct automatic MT evaluation by determining the orthographic similarity between a back-translation and the original source text. This way we eliminate the need for human translated reference texts. By correlating BLEU and back-translation scores with human judgments, it could be shown that the back-translation score gives an improved performance at the sentence level.

1 Introduction

The manual evaluation of the results of machine translation systems requires considerable time and effort. For this reason fast and inexpensive automatic methods were developed. They are based on the comparison of a machine translation with a reference translation produced by humans. The comparison is done by determining the number of matching word sequences between both translations. It could be shown that such methods, of which BLEU (Papineni et al., 2002) is the most common, can deliver evaluation results that show a high agreement with human judgments (Papineni et al., 2002; Coughlin, 2003; Koehn & Monz, 2006).

Disadvantages of BLEU and related methods are that a human reference translation is required, and that the results are reliable only at corpus level, i.e. when computed over many sentence pairs (see e.g. Callison-Burch et al., 2006). However, at the sentence level, due to data sparseness the results tend to be unsatisfactory (Agarwal & Lavie, 2008; Callison-Burch et al., 2008). Papineni et al. (2002) describe this as follows: "BLEU's strength is that it correlates highly with human judgments by averaging out individual sentence judgment errors over a test corpus rather than attempting to divine the exact human judgment for every sentence: *quantity leads to quality.*"

Although in many scenarios the above mentioned drawbacks may not be a major problem, it is nevertheless desirable to overcome them. This is what we attempt in this paper by introducing the *back-translation score*. It is based on the assumption that the MT system considered can translate a language pair in both directions, which is usually the case. Evaluating the quality of a machine translation now involves translating it back to the source language. The score is then computed by comparing the back-translation to the original source text. Although for this comparison BLEU could be used, our experiments show that a modified version which we call *OrthoBLEU* is better suited for this purpose as it can deal with compounds and inflexional variants in a more appropriate way. Its operation is based on finding matches of character- rather than word-sequences. It resembles algorithms used in translation memory search for locating orthographically similar sentences.

The results that we obtain in this work refute to some extend the common belief that back-translation (sometimes also called round-trip translation) is not a suitable means for MT evaluation (Somers, 2005; Koehn, 2005). This belief seems to be largely based on the obvious observation that the back-translation score is highest for a trivial translation system that does nothing and simply leaves all source words in place. On the other hand, according to Somers (2005) "until now no one as far as we know has published results demonstrating this" (i.e. that back-translation is not useful for MT evaluation).

We would like to add that so far the inappropriateness of back-translation has only been shown by comparisons with other automatic metrics (Somers 2005; Koehn, 2005), which are also

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 133–136,
Suntec, Singapore, 4 August 2009.

flawed. Somers (2005) therefore states: "To be really sure of our results, we should like to replicate the experiments evaluating the translations using a more old-fashioned method involving human ratings of intelligibility." That is, apparently nobody has ever seriously compared back-translation scores to human judgments, so the belief about their inutility seems not sufficiently backed by facts. This is a serious deficit which we try to overcome in this work.

2 Procedure

As our test corpus we use the first 100 English and German sentences of the *News Corpus* which was kindly provided by the organizers of the *Third Workshop on Statistical Machine Translation* (Callison-Burch et al., 2008). This corpus comprises human translations of articles from various news websites. In the case of the 100 sentences used here, the source language was Hungarian and the translations to English and German were produced from the Hungarian original. As MT evaluation is often based on multilingual corpora, the use of indirect translations appears to be a realistic scenario.

The 100 English sentences were translated to German using the online MT-system Babel Fish (`http://de.babelfish.yahoo.com/`) which is based on Systran technology. Subsequently, the translations were back-translated to English. Table 1 shows a sample sentence and its translations.

English (source)	The skyward zoom in food prices is the dominant force behind the speed up in eurozone inflation.
German (human translation)	Hauptgrund für den in der Eurozone gemessenen Anstieg der Inflation seien die rasant steigenden Lebensmittelpreise.
German (Babel Fish)	Die gen Himmel Lebensmittelpreise laut summen innen ist die dominierende Kraft hinter beschleunigen in der Eurozoneinflation.
English (back-translation)	Towards skies the food prices loud hum inside are dominating Kraft behind accelerate in the euro zone inflation.

Table 1: Sample sentence, its human translation, and its Babel Fish forward and backward translations.

The Babel Fish translations to German were judged by the author according to the standard criteria of *fluency* and *adequacy*. Hereby the scale provided by Koehn & Monz (2006) was used which assigns values between 1 and 5. We then for each sentence computed the mean of its fluency and adequacy values. This somewhat arbitrary measure serves the purposes of designating each sentence a single value, which makes the subsequent comparisons with automatic evaluations easier.

Having completed the human judgments, we next computed automatic judgments using the standard BLEU score. For this purpose we used the latest version (v12) of the NIST tool, which can be freely downloaded from the website `http://www.nist.gov/speech/tests/mt/`. This tool not only computes the BLEU score, but also a slightly modified variant, the so-called NIST score. Whereas the BLEU score assigns equal weights to all word sequences, the NIST score tries to take a sequence's information content into account by giving less frequent word sequences higher weights. In addition, the so-called *brevity penalty*, which tries to penalize too short translations, is computed somewhat differently, with the effect that small length differences have less impact on the overall score.

Using the NIST tool, the BLEU and NIST scores for all 100 translated sentences where computed. Hereby, the human translations were taken as reference. In addition, the BLEU and NIST scores were also computed for the back-translations, thereby using the source sentences as reference.

By doing so we must emphasize that, as described in the previous section, the BLEU score was not designed to deliver satisfactory results at the sentence level (Papineni et al., 2002), and this also applies to the closely related NIST score. On the other hand, there are no simple automatic evaluation tools that are suitable at the sentence level. Only the METEOR-System (Agarwal & Lavie, 2008) is a step in this direction. It takes into account inflexional variants and synonyms. However, it is considerably more sophisticated and is highly dependent on the underlying large scale linguistic resources.

We also think that – irrespectively of their design goals – the performance of the established BLEU and NIST scores at the sentence level is of some interest, especially as to our knowledge no other quantitative figures have been published so far. For the current work, as improved evaluation at the sentence level is one of the goals, this appears to be the only possibility to at all provide some baseline for a comparison using a well established automatic system.

In an attempt to reduce the concerns that arise from applying BLEU at the sentence level, we introduce OrthoBLEU. Like BLEU OrthoBLEU also compares a machine translation to a reference translation. However, instead of word sequences sequences of characters are considered, as proposed by Denoual & Lepage (2005). The OrthoBLEU score between two strings is com-

puted as the (relative) number of their matching triplets of characters (trigrams). Figure 1 illustrates this using the words *pineapple* and *apple pie*. As 6 out of 11 trigrams match, the resulting OrthoBLEU score is 54.5%.

The procedure illustrated in Figure 1 is not only applicable to words, but likewise to sentences, as punctuation marks, blanks, and special symbols can be treated like any other character. It is obvious that this procedure, which was originally developed for the purpose of fuzzy information retrieval, shows some tolerance with regard to inflexional variants, compounding, and derivations, which should be advantageous in the current setting. The source code of OrthoBLEU was written in C and can be freely downloaded from the following URL: `http://www.fask.uni-mainz.de/user/rapp/comtrans/`.

Using the OrthoBLEU algorithm, the evaluations previously conducted with the NIST tool were repeated. That is, both the Babel Fish translations as well as their back-translations were evaluated, whereby in the first case the human translations and in the second case the source sentences served as references.

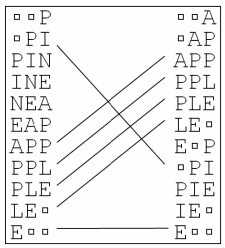


Figure 1: Computation of the OrthoBLEU score.

3 Results

Table 2 gives the average results of the evaluations described in the previous section. In columns 1 and 2 we find the human evaluation scores for fluency and adequacy, and column 3 combines them to a single score by computing their arithmetic mean. Columns 4 and 5 show the NIST and BLEU scores as computed using the NIST tool. They are based on the Babel Fish translations from English to German, whereby the human translations served as the reference. Column 6 shows the corresponding score based on OrthoBLEU, which delivers values in a range between 0% and 100%. Columns 7 to 9 show analogous scores for the back-translations. In this case the English source sentences served as the reference. As can be seen from the table, the values are higher for the back-translations. However, it would be premature to interpret this observation such that the back-translations are better suited for evaluation purposes. As these are very different tasks with different statistical properties, it would be methodologically incorrect to simply compare the absolute values. Instead we need to compute correlations between automatic and human scores.

This we did by correlating all NIST-, BLEU-, and OrthoBLEU scores for all 100 sentences with the corresponding (mean fluency/adequacy) scores from the human evaluation. We computed the Pearson product-moment correlation coefficient for all pairs, with the results being shown in Table 3. Hereby a coefficient of +1 indicates a direct linear relation, a coefficient of -1 indicates an inverse linear relation, and a coefficient of 0 indicates no linear relation.

When looking at the "translation" section of Table 3, as to be expected we obtain very low correlation coefficients for the BLEU and the NIST scores. This confirms their unsuitability for application at the sentence level as expected (see section 1). For the OrthoBLEU score we also get a very low correlation coefficient of 0.075, which means that OrthoBLEU is also unsuitable for evaluation of direct translations at the sentence level.

However, when we look at the back-translation section of Table 3, the situation is somewhat different. The correlation coefficient for the NIST score is still slightly negative, indicating that trying to take a word sequence's information content into account is hopeless at the sentence level. However, the correlation coefficient for the BLEU score almost doubles from 0.078 to 0.133, which, however, is still unsatisfactory. But a surprise comes with the OrthoBLEU score: It more than quadruples from 0.075 to 0.327, which at the sentence level is a rather good value as this result comes close to the correlation coefficient of 0.403 reported by Agarwal & Lavie (2008) as the very best of several values obtained for the METEOR system. Remember that, as described in section 2, the METEOR system requires a human-generated ref-

HUMAN EVALUATION			AUTOMATIC EVALUATION OF FORWARD-TRANSLATION			AUTOMATIC EVALUATION OF BACK-TRANSLATION		
FLUENCY	ADEQUACY	MEAN	NIST	BLEU	ORTHO-BLEU	NIST	BLEU	ORTHO-BLEU
2,49	3,06	2,78	1,31	0,01	39,72%	2,90	0,25	68,94%

Table 2: Average BLEU, NIST and OrthoBLEU scores for the 100 test sentences.

Trans-lation	Human evaluation – NIST	-0,169
	Human evaluation – BLEU	0,078
	Human evaluation – OrthoBLEU	0,075
Back-trans-lation	Human evaluation – NIST	-0,102
	Human evaluation – BLEU	0,133
	Human evaluation – OrthoBLEU	0,327

Table 3: Correlation coefficients between human and various automatic judgments based on 100 test sentences.

erence translation, large linguistic resources and comparatively sophisticated processing, and that all of this is unnecessary for the back-translation score.

4 Discussion and prospects

The motivation for this paper resulted from observing a contradiction: On one hand, practitioners sometimes recommend that (if one does not understand the target language) a back-translation can give some idea of the translation quality. Our impression has always been that this is obviously true for standard commercial systems. On the other hand, serious scientific publications (Somers, 2005; Koehn, 2005) come to the conclusion that back-translation is completely unsuitable for MT evaluation.

The outcome of the current work is in favor of the first point of view, but we should emphasize that we have no doubt about the correctness of the results presented in the publications. The discrepancy is likely to result from the following:

- The previous publications did not compare back-translation scores to human judgments but to BLEU scores only.
- The introduction of OrthoBLEU improved back-translation scores significantly.

What remains is the fact that evaluation based on back-translations can be easily fooled, e.g. by a system that does nothing, or that is capable of reversing errors. These obvious deficits have probably motivated reservations against such systems, and we agree that for such reasons they may be unsuitable for use at MT competitions.[1] However, there are numerous other applications where such considerations are of less importance. Also, it might be possible to introduce a penalty for trivial forms of translation, e.g. by counting the number of word sequences (e.g. of length 1 to 4) in a translation that are not found in a corpus of the target language.[2]

Acknowledgments

This research was in part supported by a Marie Curie Intra European Fellowship within the 7th European Community Framework Programme. We would also like to thank the anonymous reviewers for their comments, the providers of the NIST MT evaluation tool, and the organizers of the *Third Workshop on Statistical MT* for making available the *News Corpus*.

References

Abhaya Agarwal, Alon Lavie. 2008. Meteor, m-bleu and m-ter: Evaluation metrics for high-correlation with human rankings of machine translation output. *Proc. of the 3rd Workshop on Statistical MT*, Columbus, Ohio, 115–118.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Josh Schroeder. 2008. Further meta-evaluation of machine translation. *Proc. of the 3rd Workshop on Statistical MT*, Columbus, 70–106.

Chris Callison-Burch, Miles Osborne, Philipp Koehn. 2006. Re-evaluating the role of BLEU in machine translation research. *Proc. of 11th EACL*, 249–256.

Deborah Coughlin. 2003. Correlating automated and human assessments of machine translation quality. *Proc. of MT Summit IX, New Orleans*, 23–27.

Etienne Denoual, Yves Lepage. 2005. BLEU in characters: towards automatic MT evaluation in languages without word delimiters. *Proc. of 2nd IJCNLP, Companion Volume*, 81–86.

Philipp Koehn. 2005. Europarl: A parallel corpus for evaluation of machine translation. *Proceedings of the 10th MT Summit*, Phuket, Thailand, 79–86.

Philipp Koehn, Christof Monz. 2006. Manual and automatic evaluation of machine translation between European languages. *Proc. of the Workshop on Statistical MT*, New York, 102–121.

Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. Proc. of the *40th Annual Meeting of the ACL*, 311–318.

Harold Somers. 2005. Round-trip translation: what is it good for? In *Proceedings of the Australasian Language Technology Workshop ALTW 2005*. Sydney, Australia. 127–133.

[1] Although there might be a solution to this: It may not always be necessary that forward and backward translations are generated by the same MT system. For example, in an MT competition back-translations could be generated by all competing systems, and the resulting scores could be averaged.

[2] Looking up single words would not be sufficient as a system establishing any unambiguous 1:1 relationship between the source and the target language vocabulary would obtain top scores.

Sub-Sentence Division for Tree-Based Machine Translation

Hao Xiong[*], Wenwen Xu[+], Haitao Mi[*], Yang Liu[*] and **Qun Liu[*]**
[*]Key Lab. of Intelligent Information Processing
[+]Key Lab. of Computer System and Architecture
Institute of Computing Technology
Chinese Academy of Sciences
P.O. Box 2704, Beijing 100190, China
{xionghao,xuwenwen,htmi,yliu,liuqun}@ict.ac.cn

Abstract

Tree-based statistical machine translation models have made significant progress in recent years, especially when replacing 1-best trees with packed forests. However, as the parsing accuracy usually goes down dramatically with the increase of sentence length, translating long sentences often takes long time and only produces degenerate translations. We propose a new method named sub-sentence division that reduces the decoding time and improves the translation quality for tree-based translation. Our approach divides long sentences into several sub-sentences by exploiting tree structures. Large-scale experiments on the NIST 2008 Chinese-to-English test set show that our approach achieves an absolute improvement of 1.1 BLEU points over the baseline system in 50% less time.

1 Introduction

Tree-based statistical machine translation models in days have witness promising progress in recent years, such as tree-to-string models (Liu et al., 2006; Huang et al., 2006), tree-to-tree models (Quirk et al.,2005;Zhang et al., 2008). Especially, when incorporated with forest, the correspondent forest-based tree-to-string models (Mi et al., 2008; Zhang et al., 2009), tree-to-tree models (Liu et al., 2009) have achieved a promising improvements over correspondent tree-based systems. However, when we translate long sentences, we argue that two major issues will be raised. On one hand, parsing accuracy will be lower as the length of sentence grows. It will inevitably hurt the translation quality (Quirk and Corston-Oliver, 2006; Mi and Huang, 2008). On the other hand, decoding on long sentences will be time consuming, especially for forest approaches. So splitting long sentences into sub-sentences becomes a natural way in MT literature.

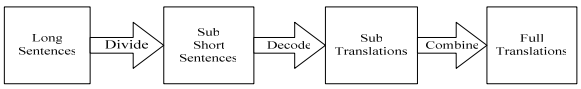


Figure 1. Main framework of our method

A simple way is to split long sentences by punctuations. However, without concerning about the original whole tree structures, this approach will result in ill-formed sub-trees which don't respect to original structures. In this paper, we present a new approach, which pays more attention to parse trees on the long sentences. We firstly parse the long sentences into trees, and then divide them accordingly into sub-sentences, which will be translated independently (Section 3). Finally, we combine sub translations into a full translation (Section 4). Large-scale experiments (Section 5) show that the BLEU score achieved by our approach is 1.1 higher than direct decoding and 0.3 higher than always splitting on commas on the 2008 NIST MT Chinese-English test set. Moreover, our approach has reduced decoding time significantly.

2 Framework

Our approach works in following steps.
(1) Split a long sentence into sub-sentences.
(2) Translate all the sub-sentences respectively.
(3) Combine the sub-translations.

Figure 1 illustrates the main idea of our approach. The crucial issues of our method are how to divide long sentences and how to combine the sub-translations.

3 Sub Sentence Division

Long sentences could be very complicated in grammar and sentence structure, thereby creating an obstacle for translation. Consequently, we need to break them into shorter and easier clauses. To divide sentences by punctuation is

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 137–140,
Suntec, Singapore, 4 August 2009.

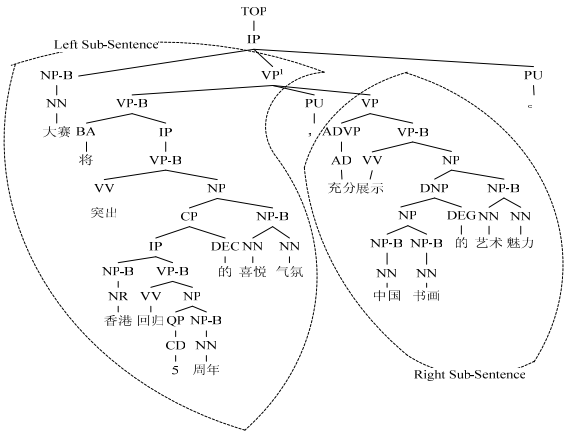


Figure 2. An undividable parse tree

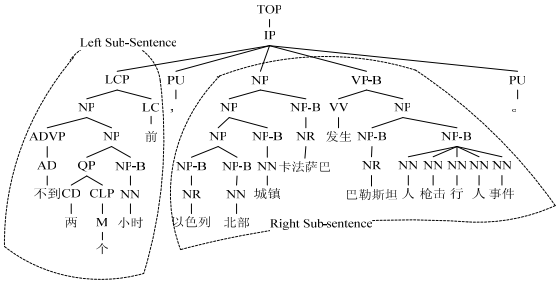


Figure 3. A dividable parse tree

one of the most commonly used methods. However, simply applying this method might damage the accuracy of parsing. As a result, the strategy we proposed is to operate division while concerning the structure of parse tree.

As sentence division should not influence the accuracy of parsing, we have to be very cautious about sentences whose division might decrease the accuracy of parsing. Figure 2(a) shows an example of the parse tree of an undividable sentence.

As can be seen in Figure 2, when we divide the sentence by comma, it would break the structure of "VP" sub-tree and result in a ill-formed sub-tree "VP" (right sub-tree), which don't have a subject and don't respect to original tree structures.

Consequently, the key issue of sentence division is finding the sentences that can be divided without loosing parsing accuracy. Figure 2(b) shows the parse tree of a sentence that can be divided by punctuation, as sub-sentences divided by comma are independent. The reference translation of the sentence in figure 3 is

Less than two hours earlier, a Palestinian took on a shooting spree on passengers in the town of Kfar Saba in northern Israel.

Pseudocode 1 Check Sub Sentence Division Algorithm

```
1: procedure CheckSubSentence(sent)
2: for each word i in sent
3:   if(i is a comma)
4:     left={words in left side of i};
       //words between last comma and current comma i
5:     right={words in right side of i};
       //words between i and next comma or semicolon, period, question mark
6:     isDividePunct[i]=true;
7:     for each j in left
8:       if(( LCA(j, i)!=parent[i])
9:         isDividePunct[i]=false;
10:        break;
11:    for each j in right
12:      if(( LCA(j, i)!=parent[i])
13:        isDividePunct[i]=false;
14:        break;
15: function LCA(i, j)
16:   return lowest common ancestor(i, j);
```

It demonstrates that this long sentence can be divided into two sub-sentences, providing a good support to our division.

In addition to dividable sentences and non-dividable sentences, there are sentences containing more than one comma, some of which are dividable and some are not. However, this does not prove to be a problem, as we process each comma independently. In other words, we only split the dividable part of this kind of sentences, leaving the non-dividable part unchanged.

To find the sentences that can be divided, we present a new method and provide its pseudo code. Firstly, we divide a sentence by its commas. For each word in the sub-sentence on the left side of a comma, we compute its lowest common ancestor (LCA) with the comma. And we process the words in the sub-sentence on the right side of the comma in the same way. Finally, we check if all the LCA we have computed are comma's parent node. If all the LCA are the comma's parent node, the sub-sentences are independent.

As shown in figure 3, the LCA (AD 不到 , PU ，), is "IP" ,which is the parent node of "PU ， "; and the LCA (NR 以色列 , PU ，) is also "IP". Till we have checked all the LCA of each word and comma, we finally find that all the LCA are "IP". As a result, this sentence can be divided without loosing parsing accuracy. LCA can be computed by using union-set (Tarjan, 1971) in lineal time. Concerning the

sub-sentence 1: 强卓指出 ***Translation 1***: Johndroe said *A1* ***Translation 2***: Johndroe pointed out *A2* ***Translation 3***: Qiang Zhuo said *A3*
comma 1: , ***Translation***: punctuation translation (white space, that …)
sub-sentence 2: 两位总统也对昨日签署的美国—南韩自由贸易协议表示欢迎 ***Translation 1***: the two presidents also welcomed the US-South Korea free trade agreement that was signed yesterday *B1* ***Translation 2***: the two presidents also expressed welcome to the US – South Korea free trade agreement signed yesterday *B2*
comma 2: , ***Translation***: punctuation translation (white space, that …)
sub-sentence 3:并将致力确保两国国会批准此一协议。 ***Translation 1***: and would work to ensure that the congresses of both countries approve this agreement. *C1* ***Translation 2***: and will make efforts to ensure the Congress to approve this agreement of the two countries. *C2*

Table 1. Sub translation example

implementation complexity, we have reduced the problem to range minimum query problem (Bender et al., 2005) with a time complexity of $o(1)$ for querying.

Above all, our approach for sub sentence works as follows:

(1)Split a sentence by semi-colon if there is one.

(2)Parse a sentence if it contains a comma, generating k-best parses (Huang Chiang, 2005) with k=10.

(3)Use the algorithm in pseudocode 1 to check the sentence and divide it if there are more than 5 parse trees indicates that the sentence is dividable.

4 Sub Translation Combining

For sub translation combining, we mainly use the best-first expansion idea from *cube pruning* (Huang and Chiang, 2007) to combine sub-translations and generate the whole *k*-best translations. We first select the best translation from sub translation sets, and then use an interpolation language model for rescoring (Huang and Chiang, 2007).

For example, we split the following sentence "*强卓指出,两位总统也对昨日签署的美国—南韩自由贸易协议表示欢迎,并将致力确保两国国会批准此一协议。*" into three sub-sentences and generate some translations, and the results are displayed in Table 1.

As seen in Table 1, for each sub-sentence, there are one or more versions of translation. For convenience, we label the three translation versions of sub-sentence 1 as A1, A2, and A3, respectively. Similarly, B1, B2, C1, C2 are also labels of translation. We push the A1, white space, B1, white space, C1 into the cube, and then generate the final translation.

According to cube pruning algorithm, we will generate other translations until we get the best list we need. Finally, we rescore the k-best list using interpolation language model and find the best translation which is *A1 that B1 white space C1*.

Test Set	02	05	08
No Sent Division	34.56	31.26	24.53
Split by Comma	34.59	31.23	25.39
Our Approach	34.86	31.23	25.69

Table 2. BLEU results (case sensitive)

Test Set	02	05	08
No Sent Division	28 h	36 h	52 h
Split by Comma	18h	23h	29h
Our Approach	18 h	22 h	26 h

Table 3. Decoding time of our experiments
(h means hours)

5 Experiments

5.1 Data preparation

We conduct our experiments on Chinese-English translation, and use the Chinese parser of Xiong et al. (2005) to parse the source sentences. And our decoder is based on forest-based tree-to-string translation model (Mi et al. 2008).

Our training corpus consists of 2.56 million sentence pairs. Forest-based rule extractor (Mi and Huang 2008) is used with a pruning threshold p=3. And we use SRI Language Modeling Toolkit (Stolcke, 2002) to train two 5-gram language models with Kneser-Ney smoothing on the English side of the training corpus and the Xinhua portion of Gigaword corpora respectively.

We use 2006 NIST MT Evaluation test set as development set, and 2002, 2005 and 2008 NIST MT Evaluation test sets as test sets. We also use *minimum error-rate training* (Och, 2003) to tune our feature weights. We evaluate our results with *case-sensitive* BLEU-4 metric (Papineni et al., 2002). The pruning threshold *p* for parse forest in decoding time is 12.

5.2 Results

The final BLEU results are shown in Table 2, our approach has achieved a BLEU score that is 1.1 higher than direct decoding and 0.3 higher than always splitting on commas.

The decoding time results are presented in Table 3. The search space of our experiment is extremely large due to the large pruning threshold (p=12), thus resulting in a long decoding time. However, our approach has reduced the decoding time by 50% over direct decoding, and 10% over always splitting on commas.

6 Conclusion & Future Work

We have presented a new sub-sentence division method and achieved some good results. In the future, we will extend our work from decoding to training time, where we divide the bilingual sentences accordingly.

Acknowledgement

The authors were supported by National Natural Science Foundation of China, Contracts 0873167 and 60736014, and 863 State Key Project No.2006AA010108. We thank Liang Huang for his insightful suggestions.

References

Bender, Farach-Colton, Pemmasani, Skiena, Sumazin, *Lowest common ancestors in trees and directed acyclic graphs.* J. Algorithms 57(2), 75–94 (2005)

Liang Huang and David Chiang. 2005. *Better kbest Parsing.* In *Proceedings of IWPT-2005*.

Liang Huang and David Chiang. 2007. *Forest rescoring: Fast decoding with integrated language models.* In *Proceedings of ACL*.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. *Statistical syntax-directed translation with extended domain of locality.* In *Proceedings of AMTA*

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. *Statistical phrase-based translation.* In *Proceedings of HLT-NAACL 2003*, pages 127-133.

Yang Liu, Qun Liu and Shouxun Lin. 2006. *Tree-to-String alignments template for statistical machine translation.* In *Proceedings of ACL*.

Yang Liu, Yajuan Lv and Qun Liu.2009. *Improving Tree-to-Tree Translation with Packed Forests.*To appear in *Proceedings of ACL/IJCNLP*..

Daniel Marcu, Wei Wang, AbdessamadEchihabi, and Kevin Knight. 2006. *Statistical Machine Translation with syntactifiedtarget language phrases.* In *Proceedings of EMNLP*.

Haitao Mi, Liang Huang, and Qun Liu. 2008. *Forest-based translation.* In *Proceedings of ACL: HLT.*

Haitao Mi and Liang Huang. 2008. *Forest-based translation rule extraction.* In *Proceedings of EMNLP*.

Franz J. Och. 2003. *Minimum error rate training in statistical machine translation.* In *Proceedings of ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. *Bleu: a method for automatic evaluation of machine translation.* In *Proceedings of ACL, pages 311–318,*.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. *Dependency treelet translation: Syntactically informed phrasal SMT.* In *Proceedings of ACL*.

Chris Quirk and Simon Corston-Oliver. 2006. *The impact of parse quality on syntactically-informed statistical machine translation.* In *Proceedings of EMNLP*.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings* of *ICSLP*, volume 30, pages 901–904.

Georgianna Tarjan, *Depth First Search and Linear Graph Algorithms.* SIAM J. Comp. 1:2, pp. 146–160, 1972.

Deyi Xiong, Shuanglong Li, Qun Liu, and Shouxun Lin.2005. *Parsing the Penn Chinese Treebank with semantic knowledge.* In *Proceedings of IJCNLP*.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. *A tree sequence alignment-based tree-to-tree translation model.* In *Proceedings of ACL*.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw and Chew Lim Tan. 2009. *Forest-based Tree Sequence to String Translation Model*. To appear in *Proceedings of ACL/IJCNLP*

Asynchronous Binarization for Synchronous Grammars

John DeNero, Adam Pauls, and **Dan Klein**
Computer Science Division
University of California, Berkeley
{denero, adpauls, klein}@cs.berkeley.edu

Abstract

Binarization of n-ary rules is critical for the efficiency of syntactic machine translation decoding. Because the target side of a rule will generally reorder the source side, it is complex (and sometimes impossible) to find synchronous rule binarizations. However, we show that synchronous binarizations are not necessary in a two-stage decoder. Instead, the grammar can be binarized one way for the parsing stage, then rebinarized in a different way for the reranking stage. Each individual binarization considers only one monolingual projection of the grammar, entirely avoiding the constraints of synchronous binarization and allowing binarizations that are separately optimized for each stage. Compared to n-ary forest reranking, even simple target-side binarization schemes improve overall decoding accuracy.

1 Introduction

Syntactic machine translation decoders search over a space of synchronous derivations, scoring them according to both a weighted synchronous grammar and an n-gram language model. The rewrites of the synchronous translation grammar are typically flat, n-ary rules. Past work has *synchronously binarized* such rules for efficiency (Zhang et al., 2006; Huang et al., 2008). Unfortunately, because source and target orders differ, synchronous binarizations can be highly constrained and sometimes impossible to find.

Recent work has explored *two-stage* decoding, which explicitly decouples decoding into a source parsing stage and a target language model integration stage (Huang and Chiang, 2007). Because translation grammars continue to increase in size and complexity, both decoding stages require efficient approaches (DeNero et al., 2009). In this paper, we show how two-stage decoding enables independent binarizations for each stage. The source-side binarization guarantees cubic-time construction of a derivation forest, while an entirely different target-side binarization leads to efficient forest reranking with a language model.

Binarizing a synchronous grammar twice independently has two principal advantages over synchronous binarization. First, each binarization can be fully tailored to its decoding stage, optimizing the efficiency of both parsing and language model reranking. Second, the ITG constraint on non-terminal reordering patterns is circumvented, allowing the efficient application of synchronous rules that do not have a synchronous binarization. The primary contribution of this paper is to establish that binarization of synchronous grammars need not be constrained by cross-lingual reordering patterns. We also demonstrate that even simple target-side binarization schemes improve the search accuracy of forest reranking with a language model, relative to n-ary forest reranking.

2 Asynchronous Binarization

Two-stage decoding consists of parsing and language model integration. The parsing stage builds a pruned forest of derivations scored by the translation grammar only. In the second stage, this forest is reranked by an n-gram language model. We rerank derivations with *cube growing*, a lazy beam search algorithm (Huang and Chiang, 2007).

In this paper, we focus on syntactic translation with tree-transducer rules (Galley et al., 2006). These synchronous rules allow multiple adjacent non-terminals and place no restrictions on rule size or lexicalization. Two example unlexicalized rules appear in Figure 1, along with aligned and parsed training sentences that would have licensed them.

2.1 Constructing Translation Forests

The parsing stage builds a forest of derivations by parsing with the source-side projection of the synchronous grammar. Each forest node $\mathbf{P}_{ij}$ compactly encodes all parse derivations rooted by grammar symbol P and spanning the source sentence from positions i to j. Each derivation of $\mathbf{P}_{ij}$ is rooted by a rule with non-terminals that each

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 141–144,
Suntec, Singapore, 4 August 2009.

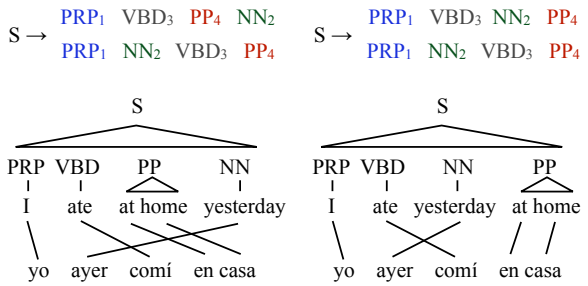


Figure 1: Two unlexicalized transducer rules (top) and aligned, parsed training sentences from which they could be extracted (bottom). The internal structure of English parses has been omitted, as it is irrelevant to our decoding problem.

anchor to some child node $\mathbf{C}_{k\ell}^{(t)}$, where the symbol $C^{(t)}$ is the tth child in the source side of the rule, and $i \leq k < \ell \leq j$.

We build this forest with a CKY-style algorithm. For each span (i, j) from small to large, and each symbol P, we iterate over all ways of building a node $\mathbf{P}_{ij}$, first considering all grammar rules with parent symbol P and then, for each rule, considering all ways of anchoring its non-terminals to existing forest nodes. Because we do not incorporate a language model in this stage, we need only operate over the source-side projection of the grammar.

Of course, the number of possible anchorings for a rule is exponential in the number of non-terminals it contains. The purpose of binarization during the parsing pass is to make this exponential algorithm polynomial by reducing rule branching to at most two non-terminals. Binarization reduces algorithmic complexity by eliminating redundant work: the shared substructures of n-ary rules are scored only once, cached, and reused. Caching is also commonplace in Early-style parsers that implicitly binarize when applying n-ary rules.

While any binarization of the source side will give a cubic-time algorithm, the particulars of a grammar transformation can affect parsing speed substantially. For instance, DeNero et al. (2009) describe normal forms particularly suited to transducer grammars, demonstrating that well-chosen binarizations admit cubic-time parsing algorithms while introducing very few intermediate grammar symbols. Binarization choice can also improve monolingual parsing efficiency (Song et al., 2008).

The parsing stage of our decoder proceeds by first converting the source-side projection of the translation grammar into lexical normal form (DeNero et al., 2009), which allows each rule to be applied to any span in linear time, then building a binary-branching translation forest, as shown in Figure 2(a). The intermediate nodes introduced during this transformation do not have a target-side projection or interpretation. They only exist for the sake of source-side parsing efficiency.

2.2 Collapsing Binarization

To facilitate a change in binarization, we transform the translation forest into n-ary form. In the n-ary forest, each hyperedge corresponds to an original grammar rule, and all nodes correspond to *original grammar symbols*, rather than those introduced during binarizaiton. Transforming the entire forest to n-ary form is intractable, however, because the number of hyperedges would be exponential in n. Instead, we include only the top k n-ary backtraces for each forest node. These backtraces can be enumerated efficiently from the binary forest. Figure 2(b) illustrates the result.

For efficiency, we follow DeNero et al. (2009) in pruning low-scoring nodes in the n-ary forest under the weighted translation grammar. We use a max-marginal threshold to prune unlikely nodes, which can be computed through a max-sum semiring variant of inside-outside (Goodman, 1996; Petrov and Klein, 2007).

Forest reranking with a language model can be performed over this n-ary forest using the cube growing algorithm of Huang and Chiang (2007). Cube growing lazily builds k-best lists of derivations at each node in the forest by filling a node-specific priority queue upon request from the parent. N-ary forest reranking serves as our baseline.

2.3 Reranking with Target-Side Binarization

Zhang et al. (2006) demonstrate that reranking over binarized derivations improves search accuracy by better exploring the space of translations within the strict confines of beam search. Binarizing the forest during reranking permits pairs of adjacent non-terminals in the target-side projection of rules to be rescored at intermediate forest nodes. This target-side binarization can be performed on-the-fly: when a node $\mathbf{P}_{ij}$ is queried for its k-best list, we binarize its n-ary backtraces.

Suppose $\mathbf{P}_{ij}$ can be constructed from a rule r with target-side projection

$$P \rightarrow \ell_0\ C_1\ \ell_1\ C_2\ \ell_2\ \ldots C_n\ \ell_n$$

where $C_1, \ldots, C_n$ are non-terminal symbols that are each anchored to a node $\mathbf{C}_{kl}^{(i)}$ in the forest, and ℓ_i are (possibly empty) sequences of lexical items.

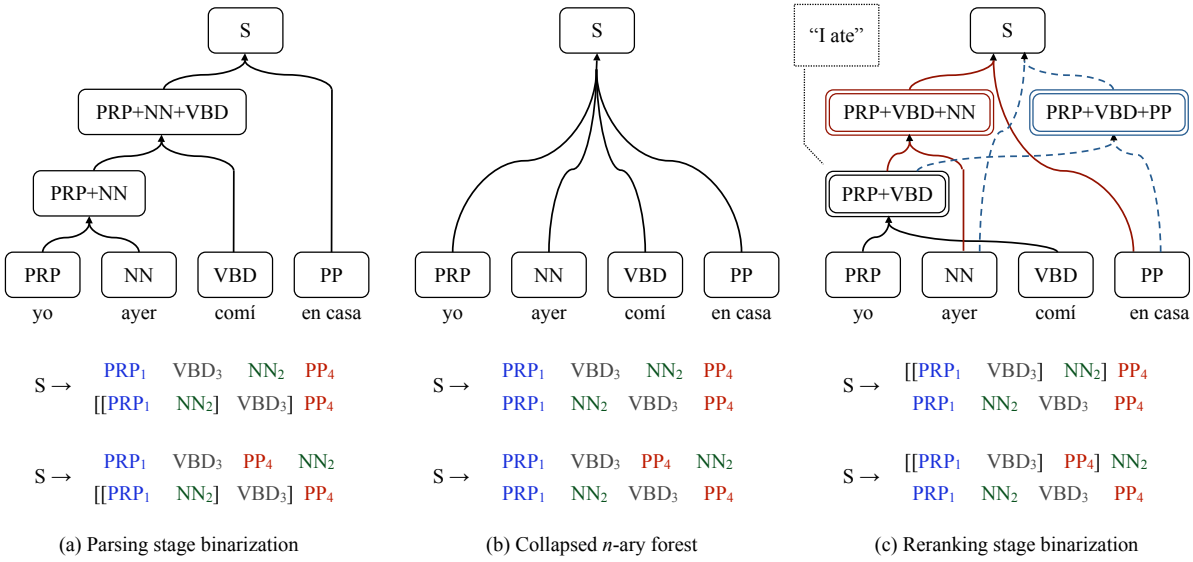


Figure 2: A translation forest as it evolves during two-stage decoding, along with two n-ary rules in the forest that are rebinarized. (a) A source-binarized forest constructed while parsing the source sentence with the translation grammar. (b) A flat n-ary forest constructed by collapsing out the source-side binarization. (c) A target-binarized forest containing two derivations of the root symbol—the second is dashed for clarity. Both derivations share the node PRP+VBD, which will contain a single k-best list of translations during language model reranking. One such translation of PRP+VBD is shown: "I ate".

We apply a simple left-branching binarization to r, though in principle any binarization is possible. We construct a new symbol B and two new rules:

$$r_1 : \; B \rightarrow \ell_0 \; C_1 \; \ell_1 \; C_2 \; \ell_2$$
$$r_2 : \; P \rightarrow B \; C_3 \; \ell_3 \ldots C_n \; \ell_n$$

These rules are also anchored to forest nodes. Any C_i remains anchored to the same node as it was in the n-ary forest. For the new symbol B, we introduce a new forest node $\mathbf{B}$ that does not correspond to any particular span of the source sentence. We likewise transform the resulting r_2 until all rules have at most two non-terminal items. The original rule r from the n-ary forest is replaced by binary rules. Figure 2(c) illustrates the rebinarized forest.

Language model reranking treats the newly introduced forest node $\mathbf{B}$ as any other node: building a k-best derivation list by combining derivations from $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$ using rule r_1. These derivations are made available to the parent of $\mathbf{B}$, which may be another introduced node (if more binarization were required) or the original root $\mathbf{P}_{ij}$.

Crucially, the ordering of non-terminals in the source-side projection of r does not play a role in this binarization process. The intermediate nodes $\mathbf{B}$ may comprise translations of discontiguous parts of the source sentence, as long as those parts are contained within the span (i, j).

2.4 Reusing Intermediate Nodes

The binarization we describe transforms the forest on a rule-by-rule basis. We must consider individual rules because they may contain different lexical items and non-terminal orderings. However, two different rules that can build a node often share some substructures. For instance, the two rules in Figure 2 both begin with PRP followed by VBD. In addition, these symbols are anchored to the same source-side spans. Thus, binarizing both rules yields the same intermediate forest node $\mathbf{B}$.

In the case where two intermediate nodes share the same intermediate rule anchored to the same forest nodes, they can be shared. That is, we need only generate one k-best list of derivations, then use it in derivations rooted by both rules. Sharing derivation lists in this way provides an additional advantage of binarization over n-ary forest reranking. Not only do we assess language model penalties over smaller partial derivations, but repeated language model evaluations are cached and reused across rules with common substructure.

3 Experiments

The utility of binarization for parsing is well known, and plays an important role in the efficiency of the parsing stage of decoding (DeNero et al., 2009). The benefit of binarization for language

Forest Reranked	BLEU	Model Score
N-ary baseline	58.2	41,543
Left-branching binary	**58.5**	**41,556**

Table 1: Reranking a binarized forest improves BLEU by 0.3 and model score by 13 relative to an n-ary forest baseline by reducing search errors during forest rescoring.

model reranking has also been established, both for synchronous binarization (Zhang et al., 2006) and for target-only binarization (Huang, 2007). In our experiment, we evaluate the benefit of target-side forest re-binarization in the two-stage decoder of DeNero et al. (2009), relative to reranking n-ary forests directly.

We translated 300 NIST 2005 Arabic sentences to English with a large grammar learned from a 220 million word bitext, using rules with up to 6 non-terminals. We used a trigram language model trained on the English side of this bitext. Model parameters were tuned with MERT. Beam size was limited to 200 derivations per forest node.

Table 1 shows a modest increase in model and BLEU score from left-branching binarization during language model reranking. We used the same pruned n-ary forest from an identical parsing stage in both conditions. Binarization did increase reranking time by 25% because more k-best lists are constructed. However, reusing intermediate edges during reranking binarization reduced binarized reranking time by 37%. We found that on average, intermediate nodes introduced in the forest are used in 4.5 different rules, which accounts for the speed increase.

4 Discussion

Asynchronous binarization in two-stage decoding allows us to select an appropriate grammar transformation for each language. The source transformation can optimize specifically for the parsing stage of translation, while the target-side binarization can optimize for the reranking stage.

Synchronous binarization is of course a way to get the benefits of binarizing both grammar projections; it is a special case of asynchronous binarization. However, synchronous binarization is constrained by the non-terminal reordering, limiting the possible binarization options. For instance, none of the binarization choices used in Figure 2 on either side would be possible in a synchronous binarization. There are rules, though rare, that cannot be binarized synchronously at all (Wu, 1997), but can be incorporated in two-stage decoding with asynchronous binarization.

On the source side, these limited binarization options may, for example, prevent a binarization that minimizes intermediate symbols (DeNero et al., 2009). On the target side, the speed of forest reranking depends upon the degree of reuse of intermediate k-best lists, which in turn depends upon the manner in which the target-side grammar projection is binarized. Limiting options may prevent a binarization that allows intermediate nodes to be maximally reused. In future work, we look forward to evaluating the wide array of forest binarization strategies that are enabled by our asynchronous approach.

References

John DeNero, Mohit Bansal, Adam Pauls, and Dan Klein. 2009. Efficient parsing for transducer grammars. In *Proceedings of the Annual Conference of the North American Association for Computational Linguistics*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the Annual Conference of the Association for Computational Linguistics*.

Joshua Goodman. 1996. Parsing algorithms and metrics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the Annual Conference of the Association for Computational Linguistics*.

Liang Huang, Hao Zhang, Daniel Gildea, and Kevin Knight. 2008. Binarization of synchronous context-free grammars. *Computational Linguistics*.

Liang Huang. 2007. Binarization, synchronous binarization, and target-side binarization. In *Proceedings of the HLT-NAACL Workshop on Syntax and Structure in Statistical Translation (SSST)*.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Xinying Song, Shilin Ding, and Chin-Yew Lin. 2008. Better binarization for the CKY parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23:377–404.

Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. 2006. Synchronous binarization for machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Hidden Markov Tree Model in Dependency-based Machine Translation*

Zdeněk Žabokrtský
Charles University in Prague
Institute of Formal and Applied Linguistics
zabokrtsky@ufal.mff.cuni.cz

Martin Popel
Charles University in Prague
Institute of Formal and Applied Linguistics
popel@matfyz.cz

Abstract

We would like to draw attention to Hidden Markov Tree Models (HMTM), which are to our knowledge still unexploited in the field of Computational Linguistics, in spite of highly successful Hidden Markov (Chain) Models. In dependency trees, the independence assumptions made by HMTM correspond to the intuition of linguistic dependency. Therefore we suggest to use HMTM and tree-modified Viterbi algorithm for tasks interpretable as labeling nodes of dependency trees. In particular, we show that the transfer phase in a Machine Translation system based on tectogrammatical dependency trees can be seen as a task suitable for HMTM. When using the HMTM approach for the English-Czech translation, we reach a moderate improvement over the baseline.

1 Introduction

Hidden Markov Tree Models (HMTM) were introduced in (Crouse et al., 1998) and used in applications such as image segmentation, signal classification, denoising, and image document categorization, see (Durand et al., 2004) for references.

Although Hidden Markov Models belong to the most successful techniques in Computational Linguistics (CL), the HMTM modification remains to the best of our knowledge unknown in the field.

The first novel claim made in this paper is that the independence assumptions made by Markov Tree Models can be useful for modeling syntactic trees. Especially, they fit dependency trees well, because these models assume conditional dependence (in the probabilistic sense) only along tree edges, which corresponds to intuition behind dependency relations (in the linguistic sense) in dependency trees. Moreover, analogously to applications of HMM on sequence labeling, HMTM can be used for labeling nodes of a dependency tree, interpreted as revealing the hidden states[1] in the tree nodes, given another (observable) labeling of the nodes of the same tree.

The second novel claim is that HMTMs are suitable for modeling the transfer phase in Machine Translation systems based on deep-syntactic dependency trees. Emission probabilities represent the translation model, whereas transition (edge) probabilities represent the target-language tree model. This decomposition can be seen as a tree-shaped analogy to the popular n-gram approaches to Statistical Machine Translation (e.g. (Koehn et al., 2003)), in which translation and language models are trainable separately too. Moreover, given the input dependency tree and HMTM parameters, there is a computationally efficient HMTM-modified Viterbi algorithm for finding the globally optimal target dependency tree.

It should be noted that when using HMTM, the source-language and target-language trees are required to be isomorphic. Obviously, this is an unrealistic assumption in real translation. However, we argue that tectogrammatical deep-syntactic dependency trees (as introduced in the Functional Generative Description framework, (Sgall, 1967)) are relatively close to this requirement, which makes the HMTM approach practically testable.

As for the related work, one can found a number of experiments with dependency-based MT in the literature, e.g., (Boguslavsky et al., 2004), (Menezes and Richardson, 2001), (Bojar, 2008). However, to our knowledge none of the published systems searches for the optimal target representa-

* The work on this project was supported by the grants MSM 0021620838, GAAV ČR 1ET101120503, and MŠMT ČR LC536. We thank Jan Hajič and three anonymous reviewers for many useful comments.

[1] HMTM looses the HMM's time and finite automaton interpretability, as the observations are not organized linearly. However, the terms "state" and "transition" are still used.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 145–148,
Suntec, Singapore, 4 August 2009.

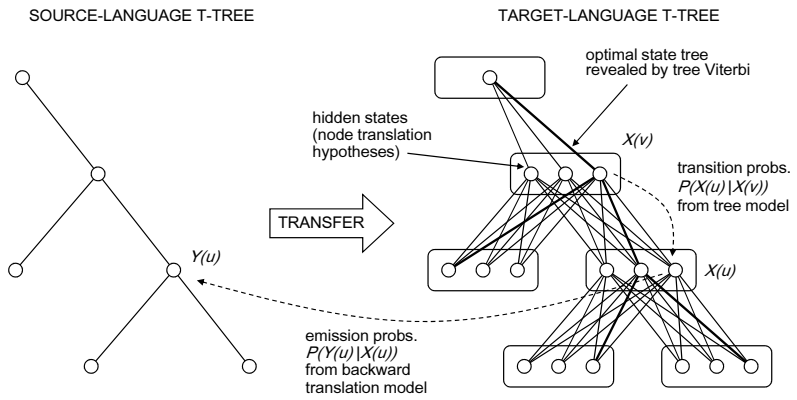


Figure 1: Tectogrammatical transfer as a task for HMTM.

tion in a way similar to HMTM.

2 Hidden Markov Tree Models

HMTM are described very briefly in this section. More detailed information can be found in (Durand et al., 2004) and in (Diligenti et al., 2003).

Suppose that $V = \{v_1, \ldots, v_{|V|}\}$ is the set of tree nodes, r is the root node and ρ is a function from $V \backslash r$ to V storing the parent node of each non-root node. Suppose two sequences of random variables, $\boldsymbol{X} = (X(v_1), \ldots, X(v_{|V|}))$ and $\boldsymbol{Y} = (Y(v_1), \ldots, Y(v_{|V|}))$, which label all nodes from V. Let $X(v)$ be understood as a hidden state of the node v, taking a value from a finite state space $S = \{s_1, \ldots, s_K\}$. Let $Y(v)$ be understood as a symbol observable on the node v, taking a value from an alphabet $K = \{k_1, \ldots, k_2\}$. Analogously to (first-order) HMMs, (first-order) HMTMs make two independence assumptions: (1) given $X(\rho(v))$, $X(v)$ is conditionally independent of any other nodes, and (2) given $X(v)$, $Y(v)$ is conditionally independent of any other nodes. Given these independence assumptions, the following factorization formula holds:[2]

$$P(\boldsymbol{Y}, \boldsymbol{X}) = P(Y(r)|X(r))P(X(r)) \cdot \prod_{v \in V \backslash r} P(Y(v)|X(v))P(X(v)|X(\rho(v))) \quad (1)$$

We see that HMTM (analogously to HMM, again) is defined by the following parameters:

- $P(X(v)|X(\rho(v)))$ – transition probabilities between the hidden states of two tree-adjacent nodes,[3]
- $P(Y(v)|X(v))$ – emission probabilities.

Naturally the question appears how to restore the most probable hidden tree labeling given the observed tree labeling (and given the tree topology, of course). As shown in (Durand et al., 2004), a modification of the HMM Viterbi algorithm can be found for HMTM. Briefly, the algorithm starts at leaf nodes and continues upwards, storing in each node for each state and each its child the optimal downward pointer to the child's hidden state. When the root is reached, the optimal state tree is retrieved by downward recursion along the pointers from the optimal root state.

3 Tree Transfer as a Task for HMTM

HMTM Assumptions from the MT Viewpoint. We suggest to use HMTM in the conventional tree-based analysis-transfer-synthesis translation scheme: (1) First we analyze an input sentence to a certain level of abstraction on which the sentence representation is tree-shaped. (2) Then we use HMTM-modified Viterbi algorithm for creating the target-language tree from the source-language tree. Labels on the source-language nodes are treated as emitted (observable) symbols, while labels on the target-language nodes are understood as hidden states which are being searched for

[2] In this work we limit ourselves to *fully stationary* HMTMs. This means that the transition and emission probabilities are independent of v. This "node invariance" is an analogy to HMM's time invariance.

[3] The need for parametrizing also $P(X(r))$ (prior probabilites of hidden states in the root node) can be avoided by adding an artificial root whose state is fixed.

(Figure 1). (3) Finally, we synthesize the target-language sentence from the target-language tree.

In the HMTM transfer step, the HMTM emission probabilities can be interpreted as probabilities from the "backward" (source given target) node-to-node translation model. HMTM transition probabilities can be interpreted as probabilities from the target-language tree model. This is an important feature from the MT viewpoint, since the decomposition into *translation model* and *language model* proved to be extremely useful in statistical MT since (Brown et al., 1993). It allows to compensate the lack of parallel resources by the relative abundance of monolingual resources.

Another advantage of the HMTM approach is that it allows us to disregard the ordering of decisions made with the individual nodes (which would be otherwise nontrivial, as for a given node there might be constraints and preferences coming both from its parent and from its children). Like in HMM, it is the notion of hidden states that facilitates "summarizing" distributed information and finding the global optimum.

On the other hand, there are several limitations implied by HMTMs which we have to consider before applying it to MT: (1) There can be only one labeling function on the source-language nodes, and one labeling function on the target-language nodes. (2) The set of hidden states and the alphabet of emitted symbols must be finite. (3) The source-language tree and the target-language tree are required to be isomorphic. In other words, only node labeling can be changed in the transfer step.

The first two assumption are easy to fulfill, but the third assumption concerning the tree isomorphism is problematic. There is no known linguistic theory guaranteeing identically shaped tree representations of a sentence and its translation. However, we would like to show in the following that the tectogrammatical layer of language description is close enough to this ideal to make the HMTM approach practically applicable.

Why Tectogrammatical Trees? Tectogrammatical layer of language description was introduced within the Functional Generative Description framework, (Sgall, 1967) and has been further elaborated in the Prague Dependency Treebank project, (Hajič and others, 2006).

On the tectogrammatical layer, each sentence is represented as a tectogrammatical tree (t-tree for short; abbreviations t-node and t-layer are used in the further text too). The main features of t-trees (from the viewpoint of our experiments) are following. Each sentence is represented as a dependency tree, whose nodes correspond to autosemantic (meaningful) words and whose edges correspond to syntactic-semantic relations (dependencies). The nodes are labeled with the lemmas of the autosemantic words. Functional words (such as prepositions, auxiliary verbs, and subordinating conjunctions) do not have nodes of their own. Information conveyed by word inflection or functional words in the surface sentence shape is represented by specialized semantic attributes attached to t-nodes (such as number or tense).

T-trees are still language specific (e.g. because of lemmas), but they largely abstract from language-specific means of expressing non-lexical meanings (such as inflection, agglutination, functional words). Next reason for using t-trees as the transfer medium is that they allow for a natural transfer factorization. One can separate the transfer into three relatively independent channels:[4] (1) transfer of lexicalization (stored in t-node's lemma attribute), (2) transfer of syntactizations (stored in t-node's formeme attribute),[5] and (3) transfer of semantically indispensable grammatical categories[6] such as number with nouns and tense with verbs (stored in specialized t-node's attributes).

Another motivation for using t-trees is that we believe that local tree contexts in t-trees carry more information relevant for correct lexical choice, compared to linear contexts in the surface sentence shapes, mainly because of long-distance dependencies and coordination structures.

Observed Symbols, Hidden States, and HMTM Parameters. The most difficult part of the tectogrammatical transfer step lies in transfer-

[4]Full independence assumption about the three channels would be inadequate, but it can be at least used for smoothing the translation probabilities.

[5]Under the term syntactization (the second channel) we understand morphosyntactic form – how the given lemma is "shaped" on the surface. We use the t-node attribute *formeme* (which is not a genuine element of the semantically oriented t-layer, but rather only a technical means that facilitates modeling the transition between t-trees and surface sentence shapes) to capture syntactization of the given t-node, with values such as n:subj – semantic noun (s.n.) in subject position, n:for+X – s.n. with preposition *for*, n:poss – possessive form of s.n., v:because+fin – semantic verb as a subordinating finite clause introduced by *because*), adj:attr – semantic adjective in attributive position.

[6]Categories only imposed by grammatical constraints (e.g. grammatical number with verbs imposed by subject-verb agreement in Czech) are disregarded on the t-layer.

ring lexicalization and syntactization (attributes lemma and formeme), while the other attributes (node ordering, grammatical number, gender, tense, person, negation, degree of comparison etc.) can be transferred by much less complex methods. As there can be only one input labeling function, we treat the following ordered pair as the observed symbol: $Y(v) = (L^{src}(v), F^{src}(v))$ where $L^{src}(v)$ is the source-language lemma of the node v and $F^{src}(v)$ is its source-language formeme. Analogously, hidden state of node v is the ordered couple $X(v) = (L^{trg}(v), F^{trg}(v))$, where $L^{trg}(v)$ is the target-language lemma of the node v and $F^{trg}(v)$ is its target-language formeme. Parameters of such HMTM are then following:

$P(X(v)|X(\rho(v))) = P(L^{trg}(v), F^{trg}(v)|L^{trg}(\rho(v)), F^{trg}(\rho(v)))$ – probability of a node labeling given its parent labeling; it can be estimated from a parsed target-language monolingual corpus, and

$P(Y(v)|X(v)) = P(L^{src}(v), F^{src}(v)|L^{trg}(v), F^{trg}(v))$ – backward translation probability; it can be estimated from a parsed and aligned parallel corpus.

To summarize: the task of tectogrammatical transfer can be formulated as revealing the values of node labeling functions L^{trg} and F^{trg} given the tree topology and given the values of node labeling functions L^{src} and F^{src}. Given the HMTM parameters specified above, the task can be solved using HMTM-modified Viterbi algorithm by interpreting the first pair as the hidden state and the second pair as the observation.

4 Experiment

To check the real applicability of HMTM transfer, we performed the following preliminary MT experiment. First, we used the tectogrammar-based MT system described in (Žabokrtský et al., 2008) as a baseline.[7] Then we substituted its transfer phase by the HMTM variant, with parameters estimated from 800 million word Czech corpus and 60 million word parallel corpus. As shown in Table 1, the HMTM approach outperforms the baseline solution both in terms of BLEU and NIST metrics.

System	BLEU	NIST
baseline system	0.0898	4.5672
HMTM modification	0.1043	4.8445

Table 1: Evaluation of English-Czech translation.

5 Conclusion

HMTM is a new approach in the field of CL. In our opinion, it has a big potential for modeling syntactic trees. To show how it can be used, we applied HMTM in an experiment on English-Czech tree-based Machine Translation and reached an improvement over the solution without HMTM.

[7] For evaluation purposes we used 2700 sentences from the evaluation section of WMT 2009 Shared Translation Task. http://www.statmt.org/wmt09/

References

Igor Boguslavsky, Leonid Iomdin, and Victor Sizov. 2004. Multilinguality in ETAP-3: Reuse of Lexical Resources. In *Proceedings of Workshop Multilingual Linguistic Resources, COLING*, pages 7–14.

Ondřej Bojar. 2008. *Exploiting Linguistic Data in Machine Translation*. Ph.D. thesis, ÚFAL, MFF UK, Prague, Czech Republic.

Peter E. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

Matthew Crouse, Robert Nowak, and Richard Baraniuk. 1998. Wavelet-based statistical signal processing using hidden markov models. *IEEE Transactions on Signal Processing*, 46(4):886–902.

Michelangelo Diligenti, Paolo Frasconi, and Marco Gori. 2003. Hidden tree Markov models for document image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:2003.

Jean-Baptiste Durand, Paulo Goncalvès, and Yann Guédon. 2004. Computational methods for hidden Markov tree models - An application to wavelet trees. *IEEE Transactions on Signal Processing*, 52(9):2551–2560.

Jan Hajič et al. 2006. Prague Dependency Treebank 2.0. Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase based translation. In *Proceedings of the HLT/NAACL*.

Arul Menezes and Stephen D. Richardson. 2001. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the workshop on Data-driven methods in machine translation*, volume 14, pages 1–8.

Petr Sgall. 1967. *Generativní popis jazyka a česká deklinace*. Academia, Prague, Czech Republic.

Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. 2008. TectoMT: Highly Modular MT System with Tectogrammatics Used as Transfer Layer. In *Proceedings of the 3rd Workshop on SMT, ACL*.

Word to Sentence Level Emotion Tagging for Bengali Blogs

Dipankar Das
Department of Computer Science & Engineering, Jadavpur University, India
dipankar.dipnil2005@gmail.com

Sivaji Bandyopadhyay
Department of Computer Science & Engineering, Jadavpur University, India
sivaji_cse_ju@yahoo.com

Abstract

In this paper, emotion analysis on blog texts has been carried out for a less privileged language like Bengali. Ekman's six basic emotion types have been selected for reliable and semi automatic word level annotation. An automatic classifier has been applied for recognizing six basic emotion types for different words in a sentence. Application of different scoring strategies to identify sentence level emotion tag based on the acquired word level emotion constituents have produced satisfactory performance.

1 Introduction

Emotion is a private state that is not open to objective observation or verification. So, the identification of the emotional state of natural language texts is really a challenging issue. Most of the related work has been conducted for English.

The approach in this paper is to assign emotion tags on the Bengali blog sentences with one of the Ekman's (1993) six basic emotion types such as *happiness, sadness, anger, fear, surprise* and *disgust*. The system consists of two phases, machine learning based word level emotion classification followed by assignment of sentence level emotion tags based on the word level constituents using sense based scoring mechanism. The classifier accuracy has been measured through confusion matrix. Corpus based and sense based *tag weights* have been calculated for each of the six emotion tags and then these emotion *tag weights* have been used to identify sentence level emotion tag. The tuned reference ranges selected from the development set have proved effective on the test set.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 briefly describes the resource preparation. Machine learning based word level emotion tagging system framework and its evaluation results have been discussed in section 4. Section 5 describes the calculation of *tag weights*, sentence level emotion detection process based on the *tag weights*, evaluation strategies and results. Finally section 6 concludes the paper.

2 Related Work

(Mishne *et al.,* 2006) used several supervised and unsupervised machine learning techniques on blog data for comparative evaluation. Importance of verbs and adjectives in identifying emotion has been explained in (Chesley *et al.,* 2006). (Yang *et al.*, 2007) has used Yahoo! Kimo Blog corpora containing emoticons associated with textual keywords to build emotion lexicons. (Chen *et al.,* 2007) has experimented the emotion classification task on web blog corpora using *Support Vector Machine* (SVM) and *Conditional Random Field* (CRF) and the observed results have shown that the CRF classifiers outperform SVM classifiers in case of document level emotion detection.

3 Resource Preparation

Bengali is a less computerized language and there is no existing emotion word list or SentiWordNet in Bengali. The English *WordNet Affect lists,* (Strapparava et al., 2004) based on Ekman's six basic emotion types have been updated with the synsets retrieved from the English *SentiWordNet* to have adequate number of emotion word entries.

These lists have been converted to Bengali using English to Bengali bilingual dictionary [1]. These six lists have been termed as *Emotion lists*. A Bengali SentiWordNet is being developed by replacing each word entry in the synonymous set of the English SentiWordNet (Esuli *et al.,* 2006)

[1] http://home.uchicago.edu/~cbs2/banglainstruction.html

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 149–152,
Suntec, Singapore, 4 August 2009.

by its equivalent Bengali meaning using the same English to Bengali bilingual dictionary.

A knowledge base for the emoticons has been prepared by experts after minutely analyzing the Bengali blog data. Each image link of the emoticon in the raw corpus has been mapped into its corresponding textual entity in the tagged corpus with the proper emotion tags using the knowledge base. The Bengali blog data have been collected from the web blog archive (www.amarblog.com) containing 1300 sentences on 14 different topics and their corresponding user comments have been retrieved.

4 Word Level Emotion Classification

Primarily, the word level annotation has been semi-automatically carried out using Ekman's six basic emotion tags. The assignment of emotion tag to a word has been done based on the type of the *Emotion Word lists* in which that word is present. Other non-emotional words have been tagged with *neutral* type. 1000 sentences have been considered for training of the CRF based word level emotion classification module. Rest 200 and 100 sentences, verified by language experts to perform evaluation have been considered as development and test data respectively.

4.1 Feature Selection and Training

The Conditional Random Field (CRF) (McCallum, 2001) framework has been used for training as well as for the classification of each word of a sentence into the above-mentioned six emotion tags and one *neutral* tag. By manually reviewing the Bengali blog data and different language specific characteristics, 10 active features have been selected heuristically for our classification task. Each feature value is boolean in nature, with discrete value for intensity feature at the word level.

- *POS information:* We are interested with the verb, noun, adjective and adverb words as these are emotion informative constituents. For this feature, total 1300 sentences has been passed through a Bengali part of speech tagger (Ekbal *et al.* 2008) based on Support Vector Machine (SVM) technique. The POS tagger was developed with a tagset of 26 POS tags[2], defined for the Indian languages. The POS tagger has demonstrated an overall accuracy of approximately 90%.

[2]http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf

- *First sentence in a topic*: It has been observed that first sentence of the topic generally contains emotion (Roth *et.al.*, 2005).
- *SentiWordNet emotion word*: A word appearing in the SentiWordNet (Bengali) contains an emotion.
- *Reduplication*: The reduplicated words (e.g., *bhallo bhallo* [good good], *khokhono khokhono* [when when] etc.) in Bengali are most likely emotion words.
- *Question words*: It has been observed that the question words generally contribute to the emotion in a sentence.
- *Colloquial / Foreign words*: The colloquial words (e.g., *kshyama* [pardon] etc.) and foreign words (e.g. Thanks, *gossya* [anger] etc.) are highly rich with their emotional contents.
- *Special punctuation symbols*: The symbols (e.g. !, ?, @ etc) appearing at the word / sentence level convey emotions.
- *Quoted sentence*: The sentences especially remarks or direct speech always contain emotion.
- *Negative word*: Negative words such as *na* (no), *noy* (not) etc. reverse the meaning of the emotion in a sentence. Such words are appropriately tagged.
- *Emoticons*: The emoticons and their consecutive occurrences generally contribute as much as real sentiment to the words or sentences that precede or follow it.

Features	Training	Testing
Parts of Speech	432	221
First Sentence	96	13
Word in SentiWordNet	684	157
Reduplication	18	7
Question Words	23	11
Coll. / Foreign Words	35	9
Special Symbols	16	4
Quoted Sentence	22	8
Negative Words	67	27
Emoticons	87	33

Table 1: Frequencies of different features

Different unigram and bi-gram context features (word level as well as POS tag level) and their combination has been generated from the training corpus. The following sentence contains four features (Colloquial word (*khyama*), special

symbol (!), quoted sentence and emotion word (ভালো [*happy*])) together and all these four features are important to identify the emotion of this sentence.

ক্ষ্যামা দাও! "তুমি ভালো লোক"
(*khyama*) (*dao*)! "(*tumi*) (*bhalo*) (lok)"
(Forgive)! "(you) (good) (person)"

4.2 Evaluation Results of the Word-level Emotion Classification

Evaluation results of the development set have demonstrated an accuracy of 56.45%. Error analysis has been conducted with the help of confusion matrix as shown in Table 2. A close investigation of the evaluation results suggests that the errors are mostly due to the uneven distribution between emotion and non-emotion tags.

Tags	*happy*	*sad*	*ang*	*dis*	*fear*	*sur*	*ntrl*
happy		0.01	0.05	0.0	0.0	0.0	0.03
sad	0.006		0.02	0.03	0.0	0.0	0.02
ang	0.0	0.03		0.0	0.02	0.0	0.01
dis	0.0	0.0	0.01		0.01	0.0	0.01
fear	0.0	0.0	0.0	0.0		0.0	0.01
sur	0.02	0.007	0.0	0.0	0.0		0.01
ntrl	0.0	0.0	0.0	0.0	0.0	0.0	

Table 2: Confusion matrix for development set

The number of non-emotional or *neutral* type tags is comparatively higher than other emotional tags in a sentence. So, one solution to this unbalanced class distribution is to split the 'non-emotion' (emo_ntrl) class into several subclasses. That is, given a POS tagset *POS*, we generate new emotion classes, 'emo_ntrl-C'|C∈*POS*. We have 26 sub-classes, which correspond, to non-emotion tags such as 'emo_ntrl-NN' (common noun), 'emo_ntrl-VFM' (verb finite main) etc. Evaluation results of the system with the inclusion of this class splitting technique have shown the accuracies of 64.65% and 66.74% on the development and test data respectively.

5 Sentence Level Emotion Tagging

This module has been developed to identify sentence level emotion tags based on the word level emotion tags.

5.1 Calculation of Emotion Tag weights

Sense_Tag_Weight (***STW***): The *tag weight* has been calculated using *SentiWordNet*. We have selected the basic six words "*happy*", "*sad*", "*anger*", "*disgust*", "*fear*" "*surprise*" as the *seed words* corresponding to each emotion type. The *positive* and *negative* scores in the English *SentiWordNet* for each synset in which each of these *seed words* appear have been retrieved and the average of the scores has been fixed as the *Sense_Tag_Weight* of that particular emotion tag.

Corpus_Tag_Weight (***CTW***): This *tag weight* for each emotion tag has been calculated based on the frequency of occurrence of an emotion tag with respect to the total number of occurrences of all six types of emotion tags in the annotated corpus.

Tag Types	*CTW*	*STW*
emo_happy	0.5112	0.0125
emo_sad	0.2327	(-) 0.1022
emo_ang	0.0959	(-) 0.5
emo_dis	0.1032	(-) 0.075
emo_fear	0.0465	0.0131
emo_sur	0.0371	0.0625
emo_ntrl	0.0	0.0

Table 3: *CTW* and *STW* for each of six emotion tags with neutral tag

5.2 Scoring Techniques

The following two scoring techniques depending on two calculated *tag weights* (in section 5.1) have been adopted for selecting the best possible sentence level emotion tags.

(1) *Sense_Weight_Score* (***SWS***): Each sentence is assigned a *Sense_Weight_Score* (***SWS***) for each emotion tag which is calculated by dividing the total *Sense_Tag_Weight* (***STW***)of all occurrences of an emotion tag in the sentence by the total *Sense_Tag_Weight* (***STW***) of all types of emotion tags present in that sentence. The *Sense_Weight_Score* is calculated as

$$SWS_i = (STWi * Ni) / (\sum_{j=1 \text{ to } 7} STW_j * Nj) \mid i \in j$$

where SWS_i is the Sentence level *Sense_Weight_Score* for the emotion tag ***i*** in the sentence and N_i is the number of occurrences of that emotion tag in the sentence. STW_i and STW_j are the *Sense_Tag_Weights* for the emotion tags ***i*** and ***j*** respectively. Each sentence has been assigned with the sentence level emotion tag SET_i for which SWS_i is highest, i.e.,

$$SET_i = [\max_{i=1 \text{ to } 6}(SWS_i)].$$

(2) *Corpus_Weight_Score* (***CWS***): This measure is calculated in a similar manner by using the ***CTW*** of each emotion tag. The corresponding Bengali sentence is assigned with the emotion tag for which the sentence level ***CWS*** is highest. The scoring mechanism has been considered for verifying any domain related biasness of emotion and their influence in emotion detection process.

5.3 Evaluation Results of Sentence Level Emotion Tagging

Each sentence in the development and test sets have been annotated *with positive* or *negative* or *neutral* valence and with any of the six emotion tags. The ***SWS*** has been used in identifying valence scores as there is no valence information carried by ***CWS***. The sentences for which the total ***SWS*** produced *positive*, *negative* and zero (0) values have been tagged as *positive, negative* and *neutral* type. Any domain biasness through ***CWS*** has been re-evaluated through ***SWS*** also. We have taken the Bengali corpus from comic related background. So, during analysis on the development set, the ***CWS*** outperforms the ***SWS*** significantly in identifying *happy, disgust, fear* and *surprise* sentence level emotion tags. The other ***SET*s** have been identified through ***SWS*** as the ***CWS*** for these ***SET*s** are significantly less than their corresponding ***SWS*** as shown in Table 5. The knowledge and information of the reference ranges (shown in Table 4) of ***SWS*** and ***CWS*** for assigning valence and six other emotion tags, acquired after tuning of development set, have been applied on the test set. The valence and emotion tag assignment process has been evaluated using accuracy measure on test data. The difference in the accuracies for the development and test sets is negligible. It signifies that the best possible reference range for valence and other emotion tags have been selected. Results in Table 5 show that the system has performed satisfactorily for valence identification as well as for sentence level emotion tagging.

Category	Reference Range
Valence (SWS)	0 to 2.35 (+ve), 0 to -0.56 (-ve) and 0.0 neutral)
happy	0.31 to 1 (CWS)
sad	-0.15 to -1.6 (SWS)
angry	-0.5 to -1.9 (SWS)
disgust	0.18 to 1 (CWS)
fear	0.14 to 1.9 (CWS)
surprise	0.15 to 1.76 (CWS)

Table 4: Reference ranges

6 Conclusion

The hierarchical ordering of the word level to sentence level and from sentence level to document level can be considered as the well favored route to track the document level emotional orientation. The handling of negative words and metaphors and their impact in detecting sentence level emotion along with document level analysis are the future areas to be explored.

Category	Development			Test
	Before		After	
	CWS	SWS		
Valence	--	**49.56**	**65.43**	**66.54**
happy	**54.15**	10.33	63.88	64.28
sad	7.66	**42.93**	64.56	66.42
angry	15.47	**53.44**	61.48	60.28
disgust	**60.13**	17.18	70.19	72.18
fear	**55.57**	11.54	66.04	67.14
surprise	**50.25**	12.39	65.45	66.45

Table 5: Accuracies (in %) of valence and six emotion tags in development set before and after applying the reference range and in test set

References

Andrea Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining.*LREC-06*.

Andrew McCallum, Fernando Pereira and John Lafferty. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and labeling Sequence Data. *ISBN*, 282 – 289.

A. Ekbal and S. Bandyopadhyay. 2008. Web-based Bengali News Corpus for Lexicon Development and POS Tagging. *POLIBITS*, 37(2008):20-29. Mexico.

B. Vincent, L. Xu, P. Chesley and R. K. Srhari. 2006. Using verbs and adjectives to automatically classify blog sentiment.*AAAI-CAAW-06*.

Carlo Strapparava, Rada Mihalcea .2007. SemEval-2007 Task 14: Affective Text. *45th Aunual Meeting of ACL*.

C. Yang, K. H.-Y. Lin, and H.-H. Chen. 2007. Building Emotion Lexicon from Weblog Corpora, *45th Annual Meeting of ACL*, pp. 133-136.

C. Yang, K. H.-Y. Lin, and H.-H. Chen.2007. Emotion Classification from Web Blog Corpora, *IEEE/WIC/ACM*, 275-278.

Cecilia Ovesdotter Alm, Dan Roth, Richard Sproat. 2005. Emotions from text: machine learning for text-based emotion prediction. *Human Language Technology and EMNLP*, 579-586.Canada.

G. Mishne and M. de Rijke. 2006. Capturing Global Mood Levels using Blog Posts, *AAAI*, *Spring Symposium on Computational Approaches to Analysing Weblogs*, 145-152.

Paul Ekman. 1993. Facial expression and emotion. *American Psychologist*, 48(4):384–392.

Extracting Comparative Sentences from Korean Text Documents Using Comparative Lexical Patterns and Machine Learning Techniques

Seon Yang
Department of Computer Engineering,
Dong-A University,
840 Hadan 2-dong, Saha-gu,
Busan 604-714 Korea
syang@donga.ac.kr

Youngjoong Ko
Department of Computer Engineering,
Dong-A University,
840 Hadan 2-dong, Saha-gu,
Busan 604-714 Korea
yjko@dau.ac.kr

Abstract

This paper proposes how to automatically identify Korean comparative sentences from text documents. This paper first investigates many comparative sentences referring to previous studies and then defines a set of comparative keywords from them. A sentence which contains one or more elements of the keyword set is called a comparative-sentence candidate. Finally, we use machine learning techniques to eliminate non-comparative sentences from the candidates. As a result, we achieved significant performance, an F1-score of 88.54%, in our experiments using various web documents.

1 Introduction

Comparing one entity with other entities is one of the most convincing ways of evaluation (Jindal and Liu, 2006). A comparative sentence formulates an ordering relation between two entities and that relation is very useful for many application areas. One key area is for the customers. For example, a customer can make a decision on his/her final choice about a digital camera after reading other customers' product reviews, e.g., "Digital Camera X is much cheaper than Y though it functions as good as Y!" Another one is for manufacturers. All the manufacturers have an interest in the articles saying how their products are compared with competitors' ones**.**

Comparative sentences often contain some comparative keywords. A sentence may express some comparison if it contains any comparative keywords such as '*보다* ([*bo-da*]: than)', '*가장* ([*ga-jang*]: most)', '*다르* ([*da-reu*]: different)', '*같* ([*gat*]: same)'. But many sentences also express comparison without those keywords. Similarly, although some sentences contain some keywords, they cannot be comparative sentences. By these reasons, extracting comparative sentences is not a simple or easy problem. It needs more complicated and challenging processes than only searching out some keywords for extracting comparative sentences.

Jindal and Liu (2006) previously studied to identify English comparative sentences. But the mechanism of Korean as an agglutinative language and that of English as an inflecting language have seriously different aspects. One of the greatest differences related to our work is that there are Part-of-Speech (POS) Tags for comparative and superlative in English[1], whereas, unfortunately, the POS tagger of Korean does not provide any comparative and superlative tags because the analysis of Korean comparative is much more difficult than that of English. The major challenge of our work is therefore to identify comparative sentences without comparative and superlative POS Tags.

We first survey previous studies about the Korean comparative syntax and collect the corpus of Korean comparative sentences from the Web. As we refer to previous studies and investigate real comparative sentences form the collected corpus, we can construct the set of comparative keywords and extract comparative-sentence candidates; the sentences which contain one or more element of the keyword set are called comparative-sentence candidates. Then we use some machine learning techniques to eliminate non-comparative sentences from those candidates. The final experimental results in 5-fold cross

[1] JJR: adjective and comparative, JJS: adjective and superlative, RBR: adverb and comparative, and RBS: adverb and superlative

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 153–156,
Suntec, Singapore, 4 August 2009.

validation show the overall precision of 88.68% and the overall recall of 88.40%.

The remainder of the paper is organized as follows. Section 2 describes the related work. In section 3, we explain comparative keywords and comparative-sentence candidates. In section 4, we describe how to eliminate non-comparative sentences from the candidates extracted in preceding section. Section 5 presents the experimental results. Finally, we discuss conclusions and future work in section 6

2 Related Work

We have not found any direct work on automatically extracting Korean comparative sentences. There is only one study by Jindal and Liu (2006) that is related to English. They used comparative and superlative POS tags and additional some keywords to search English comparative sentences. Then they used Class Sequential Rules and Naïve Bayesian learning method. Their experiment showed a precision of 79% and recall of 81%.

Our research is closely related to linguistics. Ha (1999) described Korean comparative constructions with a linguistic view. Oh (2003) discussed the gradability of comparatives. Jeong (2000) classified the adjective superlative by the type of measures.

Opinion mining is also related to our work. Many comparative sentences also contain the speaker's opinions and especially comparison is one of the most powerful tools for evaluation. We have surveyed many studies about opinion mining (Lee et al., 2008; Kim and Hovy, 2006; Wilson and Wiebe, 2003; Riloff and Wiebe, 2003; Esuli and Sebastiani, 2006).

Maximum Entropy Model is used in our technique. Berger et al. (1996) described Maximum Entropy approach to National Language Processing. In our experiments, we used Zhang's Maximum Entropy Model Toolkit (2004). Naïve Bayesian classifier is used to prove the performance of MEM (McCallum and Nigam (1998)).

3 Extracting Comparative-sentence Candidates

In this section, we define comparative keywords and extract comparative-sentence candidates by using those keywords.

3.1 Comparative keyword

First of all, we classify comparative sentences into six types and then we extract single comparative keywords from each type as follows:

Table 1. The six types of comparative sentences

	Type	Single-keyword Examples
1	Equality	'*같* ([*gat*]: same)'
2	Similarity	'*비슷하* ([*bi-seut-ha*]: similar)'
3	Difference	'*다르* ([*da-reu*]: different)'
4	Greater or lesser	'*보다* ([*bo-da*]: than)'
5	Superlative	'*가장* ([*ga-jang*]: most)'
6	Predicative	No single-keywords

We can easily find such keywords from the various sentences in first five types, while we cannot find any single keyword in the sentences of type 6.

Ex1) *"X 껌의 원재료는 초산비닐수지인데, Y 껌은 천연치클이다."* ([*X-gum-eui won-jae-ryo-neun cho-san-vi-nil-su-ji-in-de, Y-gum-eun cheon-yeon-chi-kl-i-da*]: Raw material of gum X is polyvinyl acetate, but that of Y is natural chicle.)[2]

And we can find many non-comparative sentences which contain some keywords. The following example (Ex2) shows non-comparative though it contains '*같* ([*gat*]: It means 'same', but it sometimes means 'think')'.

Ex2) *"내 생각엔 내일 비가 올 것 **같**아요."* ([*Nae sang-gak-en nae-il bi-ga ol geot gat-a-yo*]: I think it will rain tomorrow.)

Thus all the sentences can be divided into four categories as follows:

Table 2. The four categories of the sentences

Single-keyword	Contain	Not contain
Comparative Sentences	S1	S2
Non-comparative Sentences	S3	S4 (unconcerned group)

[2] In fact, type 6 can be sorted as non-comparative from linguistic view. But the speaker is probably saying that Y is better than X. This is very important comparative data as an opinion. Therefore, we also regard the sentences containing implicit comparison as comparative sentences

Our final goal is to find an effective method to extract S1 and S2, but single-keyword searching just outputs S1 and S3. In order to capture S2, we added long-distance-words sequences to the set of single-keywords. For example, we could extract '<는 [*neun*], 인데 [*in-de*], 은 [*eun*], 이다 [*i-da*]>' as a long-distance-words sequence from Ex1-sentence. It means that the sentence is formed as < S V but S V> in English (S: subject phrase, V: verb phrase). Thus we defined comparative keyword in this paper as follows:

Definition (comparative keyword)*: A comparative keyword is formed as a word or a phrase or a long-distance-words sequence. When a comparative keyword is contained in any sentence, the sentence is most likely to be a comparative sentence. (We will use an abbreviation 'CK'.)*

3.2 Comparative-sentence Candidates

We finally set up a total of 177 CKs by human efforts. In the previous work, Jindal and Liu (2006) defined 83 keywords and key phrases including comparative or superlative POS tags in English; they did not use any long-distance-words sequence.

Keyword searching process can detect most of comparative sentences (S1, S2 and S3)[3] from original text documents. That is, the recall is high but the precision is low. We here defined a comparative-sentence candidate as a sentence which contains one or more elements of the set of CKs. Now we need to eliminate the incorrect sentences (S3) from those captured sentences. First, we divided the set of CKs into two subsets denoted by CKL1 and CKL2 according to the precision of each keyword; we used 90% of the precision as a threshold value. The average precision of comparative-sentence candidates with a CKL1 keyword is 97.44% and they do not require any additional process. But that of comparative-sentence candidates with a CKL2 keyword is 29.34% and we decide to eliminate non-comparative sentences only from comparative sentence candidates with a CKL2 keyword.

4 Eliminating Non-comparative Sentences from the Candidates

To effectively eliminate non-comparative sentences from comparative sentence candidates with a CKL2 keyword, we employ machine learning techniques (MEM and Naïve Bayes). For feature extraction from each comparative-sentence candidate, we use continuous words sequence within the radius of 3 (the window size of 7) of each keyword in the sentence; we experimented with radius options of 2, 3, and 4 and we achieved the best performance in the radius of 3. After determining the radius, we replace each word with its POS tag; in order to reflect various expressions of each sentence, POS tags are more proper than lexical information of actual words. However, since CKs play the most important role to discriminate comparative sentences, they are represented as a combination of their actual keyword and POS tag. Thus our feature is formed as "$X \rightarrow y$". ('X' means a sequence and 'y' means a class; y_1 denotes comparative and y_2 denotes non-comparative). For instance, '*<pv etm nbn 기/pa ep ef sf >*[4] $\rightarrow y_2$' is one of the features from the sentence of *Ex2* in section 3.1.

5 Experimental Results

Three trained human annotators compiled a corpus of 277 online documents from various domains. They discussed their disagreements and they finally annotated 7,384 sentences. Table 3 shows the number of comparative sentences and non-comparative sentences in our corpus.

Table 3. The numbers of annotated sentences

Total	Comparative	Non-comparative
7,384	2,383 (32%)	5,001 (68%)

Before evaluating our proposed method, we conducted some experiments by machine learning techniques with all the unigrams of total actual words as baseline systems; they do not use any CKs. The precision, recall and F1-score of the baseline systems are shown at Table 4.

Table 4. The results of baseline systems (%)

Baseline System	Precision	Recall	F1-score
NB	35.98	91.62	51.66
MEM	78.17	63.34	69.94

The final overall results using the 5-fold cross validation are shown in Table 5 and Figure 1.

[3] As you can see in the experiment section, keyword searching captures 95.96% comparative sentences.

[4] The labels such as '*pv*', '*etm*', '*nbn*', etc. are Korean POS Tags

Table 5. The results of our proposed method (%)

Method	Precision	Recall	F1-score
CKs only	68.39	95.96	79.87
CKs + NB	85.42	88.59	86.67
CKs + MEM	***88.68***	***88.40***	***88.54***

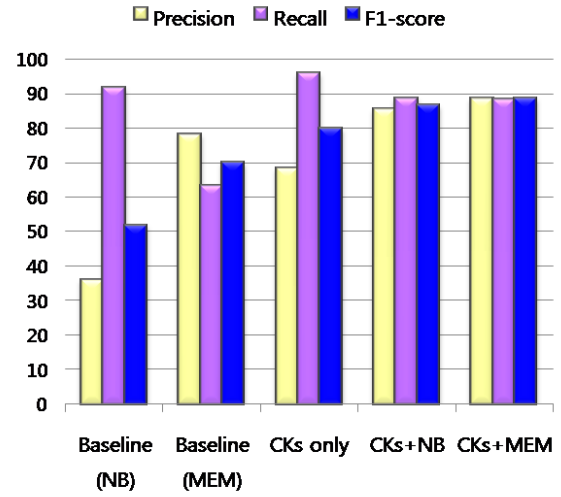


Fig. 1 The results of our proposed method (%)

As shown in Table 5 and Figure 1, both of MEM and NB is shown good performance but the F1-score of MEM is little higher than that of NB. By applying machine learning technique to our method, we can achieve high precision while we can preserve high recall.

6 Conclusions and Future Work

In this paper, we have presented how to extract comparative sentences from Korean text documents by keyword searching process and machine learning techniques. Our experimental results showed that our proposed method can be effectively used to identify comparative sentences. Since the research of comparison mining is currently in the beginning step in the world, our proposed techniques can contribute much to text mining and opinion mining research.

In our future work, we plan to classify comparative types and to extract comparative relations from identified comparative sentences.

Acknowledgement

This paper was supported by the Korean Research Foundation Grant funded by the Korean Government (KRF-2008-331-D00553)

References

Adam L. Berger et al. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39-71.

Andrea Esuli and Fabrizio Sebastiani. 2006. Determining Term Subjectivity and Term Orientation for Opinion Mining. *European Chapter of the Association for Computational Linguistics*, 193-200.

Andrew McCallum and Kamal Nigam. 1998. A Comparison of Event Models for Naïve Bayes Text Classification. *Association for Advancement of Artificial Intelligence*, 41-48.

Dong-joo Lee et al. 2008. Opinion Mining of Customer Feedback Data on the Web. *International Conference on Ubiquitous Information Management and Community*, 247-252.

Ellen Riloff and Janyce Wiebe. 2003. Learning Extraction Patterns for Subjective Expressions. *Empirical Methods in Natural Language Processing.*

Gil-jong Ha. 1999. *Korean Modern Comparative Syntax*, Pijbook Press, Seoul, Korea.

Gil-jong Ha. 1999. Research on Korean Equality Comparative Syntax, *Association for Korean Linguistics*, 5:229-265.

In-su Jeong. 2000. Research on Korean Adjective Superlative Comparative Syntax. *Korean Han-min-jok Eo-mun-hak*, 36:61-86.

Kyeong-sook Oh. 2004. The Difference between 'Man-kum' Comparative and 'Cheo-rum' Comparative. *Society of Korean Semantics*, 14:197-221.

Nitin Jindal and Bing Liu. 2006. Identifying Comparative Sentences in Text Documents, *Association for Computing Machinery/Special Interest Group on Information Retrieval*, 244-251.

Nitin Jindal and Bing Liu. 2006. Mining Comparative Sentences and Relations, *Association for Advancement of Artificial Intelligence*, 1331-1336.

Soomin Kim and Eduard Hovy. 2006. Automatic Detection of Opinion Bearing Words and Sentences. *Computational Linguistics/Association for Computational Linguistics.*

Theresa Wilson and Janyce Wiebe. 2003. Annotating Opinions in the World Press. *Special Interest Group in Discourse and Dialoque/Association for Computational Linguistics.*

Zhang Le. 2004. *Maximum Entropy Modeling Toolkit for Python and C++*. http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

Opinion and Generic Question Answering Systems: a Performance Analysis

Alexandra Balahur [1,2]
[1]DLSI, University of Alicante
Ap. De Correos 99, 03080, Alicante
[2]IPSC, EC Joint Research Centre
Via E. Fermi, 21027, Ispra
abalahur@dlsi.ua.es

Ester Boldrini
DLSI, University of Alicante
Ap. De Correos 99, 03080, Alicante
eboldrini@dlsi.ua.es

Andrés Montoyo
DLSI, University of Alicante
Ap. De Correos 99, 03080, Alicante
montoyo@dlsi.ua.es

Patricio Martínez-Barco
DLSI, University of Alicante
Ap. De Correos 99, 03080, Alicante
patricio@dlsi.ua.es

Abstract

The importance of the new textual genres such as blogs or forum entries is growing in parallel with the evolution of the Social Web. This paper presents two corpora of blog posts in English and in Spanish, annotated according to the *EmotiBlog* annotation scheme. Furthermore, we created 20 factual and opinionated questions for each language and also the *Gold Standard* for their answers in the corpus. The purpose of our work is to study the challenges involved in a mixed fact and opinion question answering setting by comparing the performance of two Question Answering (QA) systems as far as mixed opinion and factual setting is concerned. The first one is open domain, while the second one is opinion-oriented. We evaluate separately the two systems in both languages and propose possible solutions to improve QA systems that have to process mixed questions.

Introduction and motivation

In the last few years, the number of blogs has grown exponentially. Thus, the Web contains more and more subjective texts. A research from the Pew Institute shows that 75.000 blogs are created daily (Pang and Lee, 2008). They approach a great variety of topics (computer science, sociology, political science or economics) and are written by different types of people, thus are a relevant resource for large community behavior analysis. Due to the high volume of data contained in blogs, new Natural Language Processing (NLP) resources, tools and methods are needed in order to manage their language understanding. Our fist contribution consists in carrying out a multilingual research, for English and Spanish. Secondly, many sources are present in blogs, as people introduce quotes from newspaper articles or other information to support their arguments and make references to previous posts in the discussion thread. Thus, when performing a task such as Question Answering (QA), many new aspects have to be taken into consideration. Previous studies in the field (Stoyanov, Cardie and Wiebe, 2005) showed that certain types of queries, which are factual in nature, require the use of Opinion Mining (OM) resources and techniques to retrieve the correct answers. A further contribution this paper brings is the analysis and definition of the criteria for the discrimination among types of factual versus opinionated questions. Previous researchers mainly concentrated on newspaper collections. We formulated and annotated of a set of questions and answers over a multilingual *blog* collection. A further contribution is the evaluation and comparison of two different approaches to QA a fact-oriented one and another designed for opinion QA scenarios.

Related work

Research in building factoid QA systems has a long history. However, it is only recently that studies have started to focus also on the creation and development of QA systems for opinions. Recent years have seen the growth of interest in this field, both by the research performed and the publishing of various studies on the requirements

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 157–160,
Suntec, Singapore, 4 August 2009.

and peculiarities of opinion QA systems (Stoyanov, Cardie and Wiebe, 2005), (Pustejovsky and Wiebe, 2006), as well as the organization of international conferences that promote the creation of effective QA systems both for general and subjective texts, as, for example, the Text Analysis Conference (TAC)[1]. Last year's TAC 2008 Opinion QA track proposed a mixed setting of factoid ("rigid list") and opinion questions ("squishy list"), to which the traditional systems had to be adapted. The Alyssa system (Shen *et al*., 2007), classified the polarity of the question and of the extracted answer snippet, using a Support Vector Machines classifier trained on the MPQA corpus (Wiebe, Wilson and Cardie, 2005), English NTCIR[2] data and rules based on the subjectivity lexicon (Wilson, Wiebe and Hoffman, 2005). The PolyU (Wenjie *et al.,* 2008) system determines the sentiment orientation with two estimated language models for the positive versus negative categories. The QUANTA (Li, 2008) system detects the opinion holder, the object and the polarity of the opinion using a semantic labeler based on PropBank[3] and some manually defined patterns.

Evaluation

In order to carry out our evaluation, we employed a corpus of blog posts presented in (Boldrini *et al*., 2009). It is a collection of blog entries in English, Spanish and Italian. However, for this research we used the first two languages. We annotated it using *EmotiBlog* (Balahur *et al.,* 2009) and we also created a list of 20 questions for each language. Finally, we produced the *Gold Standard*, by labeling the corpus with the correct answers corresponding to the questions.

1.1 Questions

No	TYPE		QUESTION
1	F	F	What international organization do people criticize for its policy on carbon emissions? *¿Cuál fue uno de los primeros países que se preocupó por el problema medioambiental?*
2	O	F	What motivates people's negative opinions on the Kyoto Protocol? *¿Cuál es el país con mayor responsabilidad de la contaminación mundial según la opinión pública?*
3	F	F	What country do people praise for not signing the Kyoto Protocol? *¿Quién piensa que la reducción de la contaminación se debería apoyar en los consejos de los científicos?*
4	F	F	What is the nation that brings most criticism to the Kyoto Protocol? *¿Qué administración actúa totalmente en contra de la lucha contra el cambio climático?*
5	O	F	What are the reasons for the success of the Kyoto Protocol? *¿Qué personaje importante está a favor de la colaboración del estado en la lucha contra el calentamiento global?*
6	O	F	What arguments do people bring for their criticism of media as far as the Kyoto Protocol is concerned? *¿A qué políticos americanos culpa la gente por la grave situación en la que se encuentra el planeta?*
7	O	F	Why do people criticize Richard Branson? *¿A quién reprocha la gente el fracaso del Protocolo de Kyoto?*
8	F	F	What president is criticized worldwide for his reaction to the Kyoto Protocol? *¿Quién acusa a China por provocar el mayor daño al medio ambiente?*
9	F	O	What American politician is thought to have developed bad environmental policies? *¿Cómo ven los expertos el futuro?*
10	F	O	What American politician has a positive opinion on the Kyoto protocol? *Cómo se considera el atentado del 11 de septiembre?*
11	O	O	What negative opinions do people have on Hilary Benn? *¿Cuál es la opinión sobre EEUU?*
12	O	O	Why do Americans praise Al Gore's attitude towards the Kyoto protocol and other environmental issues? *¿De dónde viene la riqueza de EEUU?*
13	F	O	What country disregards the importance of the Kyoto Protocol? *¿Por qué la guerra es negativa?*
14	F	O	What country is thought to have rejected the Kyoto Protocol due to corruption? *¿Por qué Bush se retiró del Protocolo de Kyoto?*
15	F/ O	O	What alternative environmental friendly resources do people suggest to use instead of gas en the future? *¿Cuál fue la posición de EEUU sobre el Protocolo de Kyoto?*
16	F/ O	O	Is Arnold Schwarzenegger pro or against the reduction of CO2 emissions? *¿Qué piensa Bush sobre el cambio climático?*
17	F	O	What American politician supports the reduction of CO2 emissions? *¿Qué impresión da Bush?*
18	F/ O	O	What improvements are proposed to the Kyoto Protocol? *¿Qué piensa China del calentamiento global?*
19	F/ O	O	What is Bush accused of as far as political measures are concerned? *¿Cuál es la opinión de Rusia sobre el Protocolo de Kyoto?*
20	F/ O	O	What initiative of an international body is thought to be a good continuation for the Kyoto Protocol? *¿Qué cree que es necesario hacer Yvo Boer?*

Table 1: List of question in English and Spanish

As it can be seen in the table above, we created factoid (F) and opinion (O) queries for English and for Spanish; however, there are some that could be defined between factoid and opinion (F/O) and the system can retrieve multiple answers after having selected, for example, the polarity of the sentences in the corpus.

1.2 Performance of the two systems

We evaluated and compared the generic QA system of the University of Alicante (Moreda *et al.,* 2008) and the opinion QA system presented in (Balahur *et al.,* 2008), in which Named Entity Recognition with LingPipe[4] and FreeLing[5] was

[1] http://www.nist.gov/tac/

[2] http://research.nii.ac.jp/ntcir/

[3] http://verbs.colorado.edu/~mpalmer/projects/ace.html

[4] http://alias-i.com/lingpipe/

[5] http://garraf.epsevg.upc.es/freeling/

added, in order to boost the scores of answers containing NEs of the question Expected Answer Type (EAT). Table 2 presents the results obtained for English and Table 3 for Spanish. We indicate the id of the question (Q), the question type (T) and the number of answer of the *Gold Standard* (A). We present the number of the retrieved questions by the traditional system (TQA) and by the opinion one (OQA). We take into account the first 1, 5, 10 and 50 answers.

Q	T	A	Number of found answers							
			@1		@5		@10		@ 50	
			TQA	OQA	TQA	OQA	TQA	OQA	TQA	OQA
1	F	5	0	0	0	2	0	3	4	4
2	O	5	0	0	0	1	0	1	0	3
3	F	2	1	1	2	1	2	1	2	1
4	F	10	1	1	2	1	6	2	10	4
5	O	11	0	0	0	0	0	0	0	0
6	O	2	0	0	0	0	0	1	0	2
7	O	5	0	0	0	0	0	1	0	3
8	F	5	1	0	3	1	3	1	5	1
9	F	5	0	1	0	2	0	2	1	3
10	F	2	1	0	1	0	1	1	2	1
11	O	2	0	1	0	1	0	1	0	1
12	O	3	0	0	0	1	0	1	0	1
13	F	1	0	0	0	0	0	0	0	1
14	F	7	1	0	1	1	1	2	1	2
15	F/O	1	0	0	0	0	0	1	0	1
16	F/O	6	0	1	0	4	0	4	0	4
17	F	10	0	1	0	1	4	1	0	2
18	F/O	1	0	0	0	0	0	0	0	0
19	F/O	27	0	1	0	5	0	6	0	18
20	F/O	4	0	0	0	0	0	0	0	0

Table 2: Results for English

Q	T	A	Number of found answers							
			@1		@5		@10		@ 50	
			TQA	OQA	TQA	OQA	TQA	OQA	TQA	OQA
1	F	9	1	0	0	1	1	1	1	3
2	F	13	0	1	2	3	0	6	11	7
3	F	2	0	1	0	2	0	2	2	2
4	F	1	0	0	0	0	0	0	1	0
5	F	3	0	0	0	0	0	0	1	0
6	F	2	0	0	0	1	0	1	2	1
7	F	4	0	0	0	0	1	0	4	0
8	F	1	0	0	0	0	0	0	1	0
9	O	5	0	1	0	2	0	2	0	4
10	O	2	0	0	0	0	0	0	0	0
11	O	5	0	0	0	1	0	2	0	3
12	O	2	0	0	0	1	0	1	0	1
13	O	8	0	1	0	2	0	2	0	4
14	O	25	0	1	0	2	0	4	0	8
15	O	36	0	1	0	2	0	6	0	15
16	O	23	0	0	0	0	0	0	0	0
17	O	50	0	1	0	5	0	6	0	10
18	O	10	0	1	0	1	0	2	0	2
19	O	4	0	1	0	1	0	1	0	1
20	O	4	0	1	0	1	0	1	0	1

Table 3: Results for Spanish

1.3 Results and discussion

There are many problems involved when trying to perform mixed fact and opinion QA. The first can be the ambiguity of the questions e.g. *¿De dónde viene la riqueza de EEUU?*. The answer can be explicitly stated in one of the blog sentences, or a system might have to infer them from assumptions made by the bloggers and their comments. Moreover, most of the opinion questions have longer answers, not just a phrase snippet, but up to 2 or 3 sentences. As we can observe in Table 2, the questions for which the TQA system performed better were the pure factual ones (1, 3, 4, 8, 10 and 14), although in some cases (question number 14) the OQA system retrieved more correct answers. At the same time, opinion queries, although revolving around NEs, were not answered by the traditional QA system, but were satisfactorily answered by the opinion QA system (2, 5, 6, 7, 11, 12). Questions 18 and 20 were not correctly answered by any of the two systems. We believe the reason is that question 18 was ambiguous as far as polarity of the opinions expressed in the answer snippets ("improvement" does not translate to either "positive" or "negative") and question 20 referred to the title of a project proposal that was not annotated by any of the tools used. Thus, as part of the future work in our OQA system, we must add a component for the identification of quotes and titles, as well as explore a wider range of polarity/opinion scales. Furthermore, questions 15, 16, 18, 19 and 20 contain both factual as well as opinion aspects and the OQA system performed better than the TQA, although in some cases, answers were lost due to the artificial boosting of the queries containing NEs of the EAT (Expected Answer Type). Therefore, it is obvious that an extra method for answer ranking should be used, as Answer Validation techniques using Textual Entailment. In Table 3, the OQA missed some of the answers due to erroneous sentence splitting, either separating text into two sentences where it was not the case or concatenating two consecutive sentences; thus missing out on one of two consecutively annotated answers. Examples are questions number 16 and 17, where many blog entries enumerated the different arguments in consecutive sentences. Another source of problems was the fact that we gave a high weight to the presence of the NE of the sought type within the retrieved snippet and in some cases the name was misspelled in the blog entries, whereas in other NER performed by

FreeLing either attributed the wrong category to an entity, failed to annotate it or wrongfully annotated words as being NEs. Not of less importance is the question duality aspect in question 17. Bush is commented in more than 600 sentences; therefore, when polarity is not specified, it is difficult to correctly rank the answers. Finally, also the problems of temporal expressions and the coreference need to be taken into account.

Conclusions and future work

In this article, we created a collection of both factual and opinion queries in Spanish and English. We labeled the Gold Standard of the answers in the corpora and subsequently we employed two QA systems, one open domain, one for opinion questions. Our main objective was to compare the performances of these two systems and analyze their errors, proposing solutions to creating an effective QA system for both factoid an opinionated queries. We saw that, even using specialized resources, the task of QA is still challenging. Opinion QA can benefit from a snippet retrieval at a paragraph level, since in many cases the answers were not simple parts of sentences, but consisted in two or more consecutive sentences. On the other hand, we have seen cases in which each of three different consecutive sentences was a separate answer to a question. Our future work contemplates the study of the impact anaphora resolution and temporality on opinion QA, as well as the possibility to use Answer Validation techniques for answer re-ranking.

Acknowledgments

The authors would like to thank Paloma Moreda, Hector Llorens, Estela Saquete and Manuel Palomar for evaluating the questions on their QA system. This research has been partially funded by the Spanish Government under the project TEXT-MESS (TIN 2006-15265-C06-01), by the European project QALL-ME (FP6 IST 033860) and by the University of Alicante, through its doctoral scholarship.

References

Alexandra Balahur, Ester Boldrini, Andrés Montoyo, and Patricio Martínez-Barco, 2009. *Cross-topic Opinion Mining for Real-time Human-Computer Interaction.* In Proceedings of the 6th Workshop in Natural Language Processing and Cognitive Science, ICEIS 2009 Conference, Milan, Italy.

Alexandra Balahur, Elena Lloret, Oscar Ferrandez, Andrés Montoyo, Manuel Palomar, Rafael Muñoz. 2008. *The DLSIUAES Team's Participation in the TAC 2008 Tracks.* In Proceedings of the Text Analysis Conference (TAC 2008).

Ester Boldrini, Alexandra Balahur, Patricio Martínez-Barco, and Andrés Montoyo. 2009. *EmotiBlog: An Annotation Scheme for Emotion Detection and Analysis in Non-Traditional Textual Genres.* To appear in Proceedings of the 5th Conference on data Mining. Las Vegas, Nevada, USA.

W. Li, Y. Ouyang, Y. Hu, F. Wei. *PolyU at TAC 2008.* In Proceedings of Human Language Technologies Conference/Conference on Empirical methods in Natural Language Processing (HLT/EMNLP), Vancouver, BC, Canada, 2008.

Fangtao Li, Zhicheng Zheng, Tang Yang, Fan Bu, Rong Ge, Xiaoyan Zhu, Xian Zhang, and Minlie Huang. *THU QUANTA at TAC 2008 QA and RTE track.* In Proceedings of Human Language Technologies Conference/Conference on Empirical methods in Natural Language Processing (HLT/EMNLP), Vancouver, BC, Canada, 2008.

Bo Pang, and Lilian. Lee, *Opinion mining and sentiment analysis.* Foundations and Trends R. In Information Retrieval Vol. 2, Nos. 1–2 (2008) 1–135, 2008.

James Pustejovsky and Janyce. Wiebe. *Introduction to Special Issue on Advances in Question Answering.* In Language Resources and Evaluation (2005) 39: 119–122. Springer, 2006.

Dan Shen, Jochen L. Leidner, Andreas Merkel, Dietrich Klakow. *The Alyssa system at TREC QA 2007: Do we need Blog06?* In Proceedings of The Sixteenth Text Retrieval Conference (TREC 2007), Gaithersburg, MD, USA, 2007

Vaselin, Stoyanov, Claire Cardie, Janyce Wiebe. *Multi-Perspective Question Answering Using the OpQA Corpus.* In Proceedings of HLT/EMNLP. 2005.

Paloma Moreda, Hector Llorens, Estela Saquete, Manuel Palomar. 2008. Automatic Generalization of a QA Answer Extraction Module Based on Semantic Roles. *In: AAI - IBERAMIA,* Lisbon, Portugal, pages 233-242, Springer.

Janyce. Wiebe, Theresa Wilson, and Claire Cardie *Annotating expressions of opinions and emotions in language.* Language Resources and Evaluation, volume 39, issue 2-3, pp. 165-210, 2005.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. *Recognising Contextual Polarity in Phrase-level sentiment Analysis.* In Proceedings of Human language Technologies Conference/Conference on Empirical methods in Natural Language Processing (HLT/EMNLP), Vancouver, BC, Canada, 2005.

Automatic Satire Detection: Are You Having a Laugh?

Clint Burfoot
CSSE
University of Melbourne
VIC 3010 Australia
cburfoot@csse.unimelb.edu.au

Timothy Baldwin
CSSE
University of Melbourne
VIC 3010 Australia
tim@csse.unimelb.edu.au

Abstract

We introduce the novel task of determining whether a newswire article is "true" or satirical. We experiment with SVMs, feature scaling, and a number of lexical and semantic feature types, and achieve promising results over the task.

1 Introduction

This paper describes a method for filtering satirical news articles from true newswire documents. We define a satirical article as one which deliberately exposes real-world individuals, organisations and events to ridicule.

Satirical news articles tend to mimic true newswire articles, incorporating irony and non sequitur in an attempt to provide humorous insight. An example excerpt is:

> Bank Of England Governor Mervyn King is a Queen, Says Fed Chairman Ben Bernanke
>
> During last night's appearance on the American David Letterman Show, Fed Chairman Ben Bernanke let slip that Bank of England (BOE) Governor, Mervyn King, enjoys wearing women's clothing.

Contrast this with a snippet of a true newswire article:

> Delegates prepare for Cairo conference amid tight security
>
> Delegates from 156 countries began preparatory talks here Saturday ahead of the official opening of the UN World Population Conference amid tight security.

The basis for our claim that the first document is satirical is surprisingly subtle in nature, and relates to the absurdity of the suggestion that a prominent figure would expose another prominent figure as a cross dresser, the implausibility of this story appearing in a reputable news source, and the pun on the name (*King* being a *Queen*).

Satire classification is a novel task to computational linguistics. It is somewhat similar to the more widely-researched text classification tasks of spam filtering (Androutsopoulos et al., 2000) and sentiment classification (Pang and Lee, 2008), in that: (a) it is a binary classification task, and (b) it is an intrinsically semantic task, i.e. satire news articles are recognisable as such through interpretation and cross-comparison to world knowledge about the entities involved. Similarly to spam filtering and sentiment classification, a key question asked in this research is whether it is possible to perform the task on the basis of simple lexical features of various types. That is, is it possible to automatically detect satire without access to the complex inferencing and real-world knowledge that humans make use of.

The primary contributions of this research are as follows: (1) we introduce a novel task to the arena of computational linguistics and machine learning, and make available a standardised dataset for research on satire detection; and (2) we develop a method which is adept at identifying satire based on simple bag-of-words features, and further extend it to include richer features.

2 Corpus

Our satire corpus consists of a total of 4000 newswire documents and 233 satire news articles, split into fixed training and test sets as detailed in Table 1. The newswire documents were randomly sampled from the English Gigaword Corpus. The satire documents were selected to relate closely to at least one of the newswire documents by: (1) randomly selecting a newswire document; (2) hand-picking a key individual, institution or event from the selected document, and using it to formulate a phrasal query (e.g. *Bill Clinton*); (3) using the query to issue a site-restricted query to the

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 161–164,
Suntec, Singapore, 4 August 2009.

	Training	Test	Total
TRUE	2505	1495	4000
SATIRE	133	100	233

Table 1: Corpus statistics

Google search engine;[1] and (4) manually filtering out "non-newsy", irrelevant and overly-offensive documents from the top-10 returned documents (i.e. documents not containing satire news articles, or containing satire articles which were not relevant to the original query). All newswire and satire documents were then converted to plain text of consistent format using `lynx`, and all content other than the title and body of the article was manually removed (including web page menus, and header and footer data). Finally, all documents were manually post-edited to remove references to the source (e.g. *AP* or *Onion*), formatting quirks specific to a particular source (e.g. all caps in the title), and any textual metadata which was indicative of the document source (e.g. editorial notes, dates and locations). This was all in an effort to prevent classifiers from accessing superficial features which are reliable indicators of the document source and hence trivialise the satire detection process.

It is important to note that the number of satirical news articles in the corpus is significantly less than the number of true newswire articles. This reflects an impressionistic view of the web: there is far more true news content than satirical news content.

The corpus is novel to this research, and is publicly available for download at `http://www.csse.unimelb.edu.au/research/lt/resources/satire/`.

3 Method

3.1 Standard text classification approach

We take our starting point from topic-based text classification (Dumais et al., 1998; Joachims, 1998) and sentiment classification (Turney, 2002; Pang and Lee, 2008). State-of-the-art results in both fields have been achieved using support vector machines (SVMs) and bag-of-words features. We supplement the bag-of-words model with feature weighting, using the two methods described below.

Binary feature weights: Under this scheme all features are given the same weight, regardless of how many times they appear in each article. The topic and sentiment classification examples cited found binary features gave better performance than other alternatives.

Bi-normal separation feature scaling: BNS (Forman, 2008) has been shown to outperform other established feature representation schemes on a wide range of text classification tasks. This superiority is especially pronounced for collections with a low proportion of positive class instances. Under BNS, features are allocated a weight according to the formula:

$$|\mathrm{F}^{-1}(tpr) - \mathrm{F}^{-1}(fpr)|$$

where F^{-1} is the inverse normal cumulative distribution function, tpr is the true positive rate (P(feature|positive class)) and fpr is the false positive rate (P(feature|negative class)).

BNS produces the highest weights for features that are strongly correlated with either the negative or positive class. Features that occur evenly across the training instances are given the lowest weight. This behaviour is particularly helpful for features that correlate with the negative class in a negatively-skewed classification task, so in our case BNS should assist the classifier in making use of features that identify true articles.

SVM classification is performed with SVM^{light} (Joachims, 1999) using a linear kernel and the default parameter settings. Tokens are case folded; currency amounts (e.g. $2.50), abbreviations (e.g. U.S.A.), and punctuation sequences (e.g. a comma, or a closing quote mark followed by a period) are treated as separate features.

3.2 Targeted lexical features

This section describe three types of features intended to embody characteristics of satire news documents.

Headline features: Most of the articles in the corpus have a headline as their first line. To a human reader, the vast majority of the satire documents in our corpus are immediately recognisable as such from the headline alone, suggesting that our classifiers may get something out of having the

[1] The sites queried were `satirewire.com`, `theonion.com`, `newsgroper.com`, `thespoof.com`, `brokennewz.com`, `thetoque.com`, `bbspot.com`, `neowhig.org`, `humorfeed.com`, `satiricalmuslim.com`, `yunews.com`, `newsbiscuit.com`.

headline contents explicitly identified in the feature vector. To this end, we add an additional feature for each unigram appearing on the first line of an article. In this way the heading tokens are represented twice: once in the overall set of unigrams in the article, and once in the set of heading unigrams.

Profanity: true news articles very occasionally include a verbal quote which contains offensive language, but in practically all other cases it is incumbent on journalists and editors to keep their language "clean". A review of the corpus shows that this is not the case with satirical news, which occasionally uses profanity as a humorous device.

Let *P* be a binary feature indicating whether or not an article contains profanity, as determined by the `Regexp::Common::profanity` Perl module.[2]

Slang: As with profanity, it is intuitively true that true news articles tend to avoid slang. An impressionistic review of the corpus suggests that informal language is much more common to satirical articles. We measure the informality of an article as:

$$i \stackrel{\text{def}}{=} \frac{1}{|T|} \sum_{t \in T} s(t)$$

where T is the set of unigram tokens in the article and s is a function taking the value 1 if the token has a dictionary definition marked as slang and 0 if it does not.

It is important to note that this measure of "informality" is approximate at best. We do not attempt, e.g., to disambiguate the sense of individual word terms to tell whether the slang sense of a word is the one intended. Rather, we simply check to see if each word has a slang usage in Wiktionary.[3]

A continuous feature is set to the value of i for each article. Discrete features *highi* and *lowi* are set as:

$$\textit{highi} \stackrel{\text{def}}{=} \begin{cases} 1 & v > \bar{i} + 2\sigma; \\ 0 & \end{cases}$$

$$\textit{lowi} \stackrel{\text{def}}{=} \begin{cases} 1 & v < \bar{i} - 2\sigma; \\ 0 & \end{cases}$$

where $\bar{i}$ and σ are, respectively, the mean and standard deviation of i across all articles.

[2] `http://search.cpan.org/perldoc?Regexp::Common::profanity`

[3] `http://www.wiktionary.org`

3.3 Semantic validity

Lexical approaches are clearly inadequate if we assume that good satirical news articles tend to emulate real news in tone, style, and content. What is needed is an approach that captures the document semantics.

One common device in satire news articles is absurdity, in terms of describing well-known individuals in unfamiliar settings which parody their viewpoints or public profile. We attempt to capture this via *validity*, in the form of the relative frequency of the particular combination of key participants reported in the story. Our method identifies the named entities in a given document and queries the web for the conjunction of those entities. Our expectation is that true news stories will have been reported in various forums, and hence the number of web documents which include the same combination of entities will be higher than with satire documents.

To implement this method, we first use the Stanford Named Entity Recognizer[4] (Finkel et al., 2005) to identify the set of person and organisation entities, E, from each article in the corpus.

From this, we estimate the validity of the combination of entities in the article as:

$$v(E) \stackrel{\text{def}}{=} |g(E)|$$

where g is the set of matching documents returned by Google using a conjunctive query. We anticipate that v will have two potentially useful properties: (1) it will be relatively lower when E includes made-up entity names such as *Hitler Commemoration Institute*, found in one satirical corpus article; and (2) it will be relatively lower when E contains unusual combinations of entities such as, for example, those in the satirical article beginning *Missing Brazilian balloonist Padre spotted straddling Pink Floyd flying pig*.

We include both a continuous representation of v for each article, in the form of $\log(v(E))$, and discrete variants of the feature, based on the same methodology as for *highi* and *lowi*.

4 Results

The results for our classifiers over the satire corpus are shown in Table 2. The baseline is a naive classifier that assigns all instances to the positive

[4] `http://nlp.stanford.edu/software/CRF-NER.shtml`

("article⇒SATIRE?")	P	R	F
all-positive baseline	0.063	**1.000**	0.118
BIN	0.943	0.500	0.654
BIN+lex	0.945	0.520	0.671
BIN+val	0.943	0.500	0.654
BIN+all	0.945	0.520	0.671
BNS	0.944	0.670	0.784
BNS+lex	0.957	0.660	0.781
BNS+val	0.945	0.690	**0.798**
BNS+all	**0.958**	0.680	0.795

Table 2: Results for satire detection (P = precision, R = recall, and F = F-score) for binary unigram features (BIN) and BNS unigram features (BNS), optionally using lexical (lex), validity (val) or combined lexical and validity (all) features

class (i.e. SATIRE). An SVM classifier with simple binary unigram word features provides a standard text classification benchmark.

All of the classifiers easily outperform the baseline. This is to be expected given the low proportion of positive instances in the corpus. The benchmark classifier has very good precision, but recall of only 0.500. Adding the heading, slang, and profanity features provides a small improvement in both precision and recall.

Moving to BNS feature scaling keeps the very high precision and increases the recall to 0.670. Adding in the heading, slang and profanity lexical features ("+lex") actually decreases the F-score slightly, but adding the validity features ("+val") provides a near 2 point F-score increase, resulting in the best overall F-score of 0.798.

All of the BNS scores achieve statistically significant improvements over the benchmark in terms of F-score (using approximate randomisation, $p < 0.05$). The 1-2% gains given by adding in the various feature types are not statistically significant due to the small number of satire instances concerned.

All of the classifiers achieve very high precision and considerably lower recall. Error analysis suggests that the reason for the lower recall is subtler satire articles, which require detailed knowledge of the individuals to be fully appreciated as satire. While they are not perfect, however, the classifiers achieve remarkably high performance given the superficiality of the features used.

5 Conclusions and future work

This paper has introduced a novel task to computational linguistics and machine learning: determining whether a newswire article is "true" or satirical. We found that the combination of SVMs with BNS feature scaling achieves high precision and lower recall, and that the inclusion of the notion of "validity" achieves the best overall F-score.

References

Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras, and Constantine D. Spyropoulos. 2000. An evaluation of Naive Bayesian anti-spam filtering. In *Proceedings of the 11th European Conference on Machine Learning*, pages 9–17, Barcelona, Spain.

Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. 1998. Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, pages 148–155, New York, USA.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, Ann Arbor, USA.

George Forman. 2008. BNS scaling: An improved representation over TF-IDF for SVM text classification. In *Proceedings of the 17th International Conference on Information and Knowledge Management*, pages 263–270, Napa Valley, USA.

Thorsten Joachims. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, Germany.

Thorsten Joachims. 1999. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 169–184. MIT Press, Cambridge, USA.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, USA.

Hierarchical Multi-Class Text Categorization with Global Margin Maximization

Xipeng Qiu
School of Computer Science
Fudan University
xpqiu@fudan.edu.cn

Wenjun Gao
School of Computer Science
Fudan University
wjgao616@gmail.com

Xuanjing Huang
School of Computer Science
Fudan University
xjhuang@fudan.edu.cn

Abstract

Text categorization is a crucial and well-proven method for organizing the collection of large scale documents. In this paper, we propose a hierarchical multi-class text categorization method with global margin maximization. We not only maximize the margins among leaf categories, but also maximize the margins among their ancestors. Experiments show that the performance of our algorithm is competitive with the recently proposed hierarchical multi-class classification algorithms.

1 Introduction

In the past serval years, hierarchical text categorization has become an active research topic in database area (Koller and Sahami, 1997; Weigend et al., 1999) and machine learning area (Rousu et al., 2006; Cai and Hofmann, 2007).

Hierarchical categorization methods can be divided in two types: local and global approaches (Wang et al., 1999; Sun and Lim, 2001). A local approach usually proceeds in a top-down fashion, which firstly picks the most relevant categories of the top level and then recursively making the choice among the low-level categories. The global approach builds only one classifier to discriminate all categories in a hierarchy. Due that the global hierarchical categorization can avoid the drawbacks about those high-level irrecoverable error, it is more popular in the machine learning domain.

The essential idea behind global approach is that the close classes(nodes) have some common underlying factors. Especially, the descendant classes can share the characteristics of the ancestor classes, which is similar with multi-task learning(Caruana, 1997). A key problem for global hierarchical categorization is how to combine these underlying factors.

In this paper, we propose an method for hierarchical multi-class text categorization with global margin maximization. We emphasize that it is important to separate all the nodes of the correct path in the class hierarchy from their sibling node, then we incorporate such information into the formulation of hierarchical support vector machine.

The rest of the paper is organized as follows. Section 2 describes the basic model of multi-class hierarchical categorization with maximizing margin. Then we propose our improved versions in section 3. Section 4 gives the experimental analysis. Section 5 concludes the paper.

2 Hierarchical Multi-Class Text Categorization

Multiclass SVM can be generalized to the problem of hierarchical categorization (Cai and Hofmann, 2007), which has more than two categories in most of the case. Denote Y_i as the multilabels of $\mathbf{x}_i$ and $\bar{Y}_i$ the multilabels set not in Y_i. The separation margin of $\mathbf{w}$, with respect to $\mathbf{x}_i$, can be approximated as:

$$\gamma_i(\mathbf{w}) = \min_{\mathbf{y}\in Y_i, \bar{\mathbf{y}}\in\bar{Y}_i} \langle \Phi(\mathbf{x}_i, \mathbf{y}) - \Phi(\mathbf{x}_i, \bar{\mathbf{y}}), \mathbf{w}\rangle \qquad (1)$$

The loss function can be accommodated to multi-class SVM to scale the penalties for margin violations proportional to the loss. This is motivated by the fact that margin violations involving an incorrect class with high loss should be penalized more severely. So the cost-sensitive hierarchical multiclass formulation takes takes the following form:

$$\min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \qquad (2)$$
$$\text{s.t.} \langle \mathbf{w}, \delta\Phi_i(\mathbf{y},\bar{\mathbf{y}})\rangle \geq 1 - \frac{\xi_i}{l(\mathbf{y},\bar{\mathbf{y}})},\ (\forall i, \mathbf{y}\in Y_i, \bar{\mathbf{y}}\in\bar{Y}_i)$$
$$\xi_i \geq 0 (\forall i)$$

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 165–168,
Suntec, Singapore, 4 August 2009.

where $\delta\Phi_i(\mathbf{y}, \bar{\mathbf{y}}) = \Phi(\mathbf{x}_i, \mathbf{y}) - \Phi(\mathbf{x}_i, \bar{\mathbf{y}})$, $l(\mathbf{y}, \bar{\mathbf{y}}) > 0$ and $\Phi(\mathbf{x}, \mathbf{y})$ is the joint feature of input $\mathbf{x}$ and output $\mathbf{y}$, which can be represented as:

$$\Phi(\mathbf{x}, \mathbf{y}) = \Lambda(\mathbf{y}) \otimes \phi(\mathbf{x}) \tag{3}$$

where $\otimes$ is the tensor product. $\Lambda(\mathbf{y})$ is the feature representation of $\mathbf{y}$.

Thus, we can classify a document $\mathbf{x}$ to label $y^\star$:

$$y^\star = \arg\max_y F(\mathbf{w}, \Phi(\mathbf{x}, \mathbf{y})) \tag{4}$$

where $F(\cdot)$ is a map function.

There are different kinds of loss functions $l(\mathbf{y}, \bar{\mathbf{y}})$.

One is the**zero-one loss**, $l_{0/1}(\mathbf{y}, \mathbf{u}) = [\mathbf{y} \neq \mathbf{u}]$.

Another is specially designed for the hierarchy is **tree loss**(Dekel et al., 2004). Tree loss is defined as the length of the path between two multilabels with positive microlabels,

$$l_{tr} = |path(i : \mathbf{y}_i = 1, j : \mathbf{u}_j = 1)| \tag{5}$$

(Rousu et al., 2006) proposed a simplified version of l_H, namely $l_{\tilde{H}}$:

$$l_{\hat{H}} = \sum_j c_j [y_j \neq u_j \& y_{pa}(j) = u_{pa(j)}], \tag{6}$$

that penalizes a mistake in a child only if the label of the parent was correct. There are some different choices for setting c_j. One naive idea is to use a uniform weighting ($c_j = 1$). Another possible choice is to divide the loss among the sibling:

$$c_{root} = 1, c_j = c_{Parent(j)}/(|Sib(j)| + 1) \tag{7}$$

Another possible choice is to scale the loss by the proportion of the hierarchy that is in the subtree $T(j)$ rooted by j:

$$c_j = |T(j)|/|T(root)| \tag{8}$$

Using these scaling weights, the derived losses are referred as $l_{\hat{uni}}$,$l_{\hat{sib}}$ and $l_{\hat{sub}}$ respectively.

3 Hierarchical Multi-Class Text Categorization with Global Margin Maximization

In previous literature (Cai and Hofmann, 2004; Tsochantaridis et al., 2005), they focused on separating the correct path from those incorrect path. Inspired by the example in Figure 1, we emphasize it is also important to separate the ancestor node in the correct path from their sibling node.

The vector $\mathbf{w}$ can be decomposed in to the set of $\mathbf{w}_i$ for each node (category) in the hierarchy. In Figure 1, the example hierarchy has 7 nodes and 4 of them are leaf nodes. The category is encode as an integer, $1, \ldots, 7$. Suppose that the training pattern $\mathbf{x}$ belongs to category 4. Both $\mathbf{w}$ in the Figure 1a and Figure 1b can successfully classify $\mathbf{x}$ into category 4, since $F(\mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_4)) = \sum_{1,2,4} \langle \mathbf{w}_i, \mathbf{x} \rangle$ is the maximal among all the possible discriminate functions. So both learned parameter $\mathbf{w}$ is acceptable in current hierarchical support vector machine.

Here we claim the $\mathbf{w}$ in Figure 1b is better than the $\mathbf{w}$ in Figure 1a. Since we notice in Figure 1a, the discriminate function $\langle \mathbf{w}_2, \mathbf{x} \rangle$ is smaller than the discriminate function $\langle \mathbf{w}_3, \mathbf{x} \rangle$. The discriminate function $\langle \mathbf{w}_i, \mathbf{x} \rangle$ measures the similarity of $\mathbf{x}$ to category i. The larger the discriminate function is, the more similar $\mathbf{x}$ is to category i. Since category 2 is in the path from the root to the correct category and category 3 is not, intuitively, $\mathbf{x}$ should be closer to category 2 than category 3. But the discriminate function in Figure 1a is contradictive to this assumption. But such information is reflected correctly in Figure 1b. So we conclude $\mathbf{w}$ in Fig. 1b is superior to $\mathbf{w}$ in 1a.

Here we propose a novel formulation to incorporate such information. Denote A_i as the multilabel in Y_i that corresponds to the nonleaf categories and $Sib(z)$ denotes the sibling nodes of z, that is the set of nodes that have the same parent with z, except z itself. Implementing the above idea, we can get the following formulation:

$$\begin{aligned}
\min_{\mathbf{w},\xi,\zeta} \frac{1}{2}\|\mathbf{w}\|^2 \quad &+ \quad C_1 \sum_i \xi_i + C_2 \sum_i \zeta_i \qquad (9)\\
\text{s.t.} \langle \mathbf{w}, \delta\Phi_i(\mathbf{y}, \bar{\mathbf{y}}) \rangle \quad &\geq \quad 1 - \frac{\xi_i}{l(\mathbf{y}, \bar{\mathbf{y}})}, (\forall i, \begin{smallmatrix} \mathbf{y} \in Y_i \\ \bar{\mathbf{y}} \in \bar{Y}_i \end{smallmatrix})\\
\langle \mathbf{w}, \delta\Phi_i(\mathbf{z}, \bar{\mathbf{z}}) \rangle \quad &\geq \quad 1 - \frac{\zeta_i}{l(\mathbf{z}, \bar{\mathbf{z}})}, (\forall i, \begin{smallmatrix} \mathbf{z} \in A(i) \\ \bar{\mathbf{z}} \in Sib(\mathbf{z}) \end{smallmatrix})\\
\xi_i \quad &\geq \quad 0 (\forall i)\\
\zeta_i \quad &\geq \quad 0 (\forall i)
\end{aligned}$$

It arrives at the following Lagrangian:

$$\begin{aligned}
&L(\mathbf{w}, \xi_1, ..., \xi_n, \zeta_1, ..., \zeta_n)\\
&= \frac{1}{2}\|\mathbf{w}\|^2 + C_1 \sum_i \xi_i + C_2 \sum_i \zeta_i\\
&- \sum_i \sum_{\substack{\mathbf{y} \in Y_i \\ \bar{\mathbf{y}} \in \bar{Y}_i}} \alpha_{i\mathbf{y}\bar{\mathbf{y}}} (\langle \mathbf{w}, \delta\Phi_i(\mathbf{y}, \bar{\mathbf{y}}) \rangle - 1 + \frac{\xi_i}{l(\mathbf{y}, \bar{\mathbf{y}})})
\end{aligned}$$

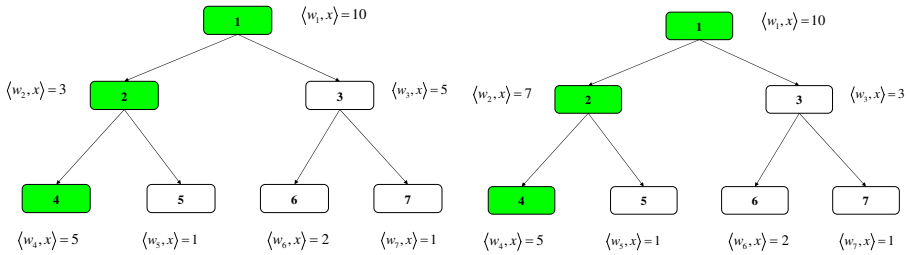


Figure 1: Two different discriminant function in a hierarchy

$$-\sum_i \sum_{\substack{\mathbf{z}\in A_i \\ \bar{\mathbf{z}}\in Sib(\mathbf{z})}} \beta_{i\mathbf{z}\bar{\mathbf{z}}}(\langle \mathbf{w}, \delta\Phi_i(\mathbf{z},\bar{\mathbf{z}})\rangle - 1 + \frac{\zeta_i}{l(\mathbf{z},\bar{\mathbf{z}})})$$
$$-\sum_i c_i\xi_i - \sum_i d_i\zeta_i \qquad (10)$$

The dual QP becomes

$$\max_{\boldsymbol{\alpha}} \Theta(\boldsymbol{\alpha}) = \sum_i \sum_{\substack{\mathbf{y}\in Y_i \\ \bar{\mathbf{y}}\in \bar{Y}_i}} \alpha_{i\mathbf{y}\bar{\mathbf{y}}} + \sum_i \sum_{\substack{\mathbf{z}\in A_i \\ \bar{\mathbf{z}}\in Sib(\mathbf{z})}} \beta_{i\mathbf{z}\bar{\mathbf{z}}}$$
$$-\frac{1}{2}\sum_{i,j}\sum_{\substack{\mathbf{y}\in Y_i \\ \bar{\mathbf{y}}\in \bar{Y}_i}}\sum_{\substack{\mathbf{r}\in Y_j \\ \bar{\mathbf{r}}\in \bar{Y}_j}} \theta^1_{i,j,\mathbf{y},\bar{\mathbf{y}},\mathbf{r},\bar{\mathbf{r}}} \qquad (11)$$
$$-\frac{1}{2}\sum_{i,j}\sum_{\substack{\mathbf{z}\in A_i \\ \bar{\mathbf{z}}\in Sib(\mathbf{z})}}\sum_{\substack{\mathbf{k}\in A_j \\ \bar{\mathbf{k}}\in Sib(\mathbf{k})}} \theta^2_{i,j,\mathbf{z},\bar{\mathbf{z}},\mathbf{k},\bar{\mathbf{k}}},$$

$$\text{s.t.}\alpha_{i\mathbf{y}\bar{\mathbf{y}}} \geq 0, \qquad (12)$$
$$\beta_{j\mathbf{z}\bar{\mathbf{z}}} \geq 0, \qquad (13)$$
$$\sum_{\substack{\mathbf{y}\in Y_i \\ \bar{\mathbf{y}}\in \bar{Y}_i}} \frac{\alpha_{i\mathbf{y}\bar{\mathbf{y}}}}{l(\mathbf{y},\bar{\mathbf{y}})} \leq C_1, \qquad (14)$$
$$\sum_{\substack{\mathbf{z}\in A_i \\ \bar{\mathbf{z}}\in Sib(\mathbf{z})}} \frac{\beta_{i\mathbf{z}\bar{\mathbf{z}}}}{l(\mathbf{z},\bar{\mathbf{z}})} \leq C_2, \qquad (15)$$

where $\theta^1_{i,j,\mathbf{y},\bar{\mathbf{y}},\mathbf{r},\bar{\mathbf{r}}} = \alpha_{i\mathbf{y}\bar{\mathbf{y}}}\alpha_{j\mathbf{r}\bar{\mathbf{r}}}\langle\delta\Phi_i(\mathbf{y},\bar{\mathbf{y}}),\delta\Phi_j(\mathbf{r},\bar{\mathbf{r}})\rangle$ and $\theta^2_{i,j,\mathbf{z},\bar{\mathbf{z}},\mathbf{k},\bar{\mathbf{k}}} = \beta_{i\mathbf{z}\bar{\mathbf{z}}}\beta_{j\mathbf{k}\bar{\mathbf{k}}}\langle\delta\Phi_i(\mathbf{z},\bar{\mathbf{z}}),\delta\Phi_j(\mathbf{k},\bar{\mathbf{k}})\rangle$.

3.1 Optimization Algorithm

The derived QP can be very large, since the number of α and β variables is up to $O(n*2^N)$, where n is number of training pattern and N is the number of nodes in the hierarchy. But two properties of the dual problem can be exploited to design a much more efficient optimization.

First, the constraints in the dual problem Eq. 11 - Eq. 15 factorize over the instance index for both α-variables and β-variables. The constraints in Eq. 14 do not couple α-variables and β-variables together. Further, dual variables $\alpha_{i\mathbf{y}\bar{\mathbf{y}}}$ and $\alpha_{j\mathbf{y}'\bar{\mathbf{y}}'}$ belonging to different training instances i and j do not join in a same constraints. This inspired an optimization procedure which iteratively performs subspace optimization over all dual variables $\alpha_{i\mathbf{y}\bar{\mathbf{y}}}$ belonging to the same training instance. This will in general reduced to a much smaller QP, since it freezes all $\alpha_{j\mathbf{y}\bar{\mathbf{y}}}$ with $j \neq i$ and β-variables at their current values. This strategy can be applied in solving β-variables.

Secondly, the number of active constraints at the solution is expected to be relatively small, since only a small fraction of categories $\bar{\mathbf{y}} \in \bar{Y}_i$ (or $\bar{\mathbf{y}} \in Sib(\mathbf{y})$ when $\mathbf{y} \in A_i$) will typically fail to achieve the required margin. The expected sparseness of the variable for the dual problem can be exploited by employing a variable selection strategy. Equivalently, this corresponds to a cutting plane algorithm for the primal QP. Intuitively, we will identify the most violated margin constraint with index $(i,\mathbf{y},\bar{\mathbf{y}})$ and then add the corresponding variable to the optimization problem. This means that we start with extremely sparse problems and only successively increase the number of variables in the active set. This general approach to deal with large linear or quadratic optimization problems is also known as column selection. In practice, it is often not necessary to optimize until final convergence, which adds to the attractiveness of this approach.

We have used the LOQO optimization package (Vanderbei, 1999) in our experiments.

4 Experiment

We evaluate our proposed model on the section D in the WIPO-alpha collection[1], which consists of the 1372 training and 358 testing document. The

[1]World Intellectual Property Organization (WIPO)

Table 1: Prediction losses (%) obtained on WIPO. The values per column is calculated with the different loss function.

Train \ Test		$l_{0/1}$	l_{Δ}	l_{tr}	l_{uni}	l_{sib}	l_{sub}
$l_{0/1}$	HSVM	48.6	188.8	94.4	97.2	5.4	7.5
	HSVM-S	**48.3**	**186.6**	**93.3**	**96.6**	**5.2**	**7.4**
l_{Δ}	HSVM	49.7	187.7	93.9	99.4	5.0	7.1
	HSVM-S	**47.8**	**165.3**	**89.7**	90.5	**4.8**	**6.9**
	HM3	70.9	167.0	-	89.1	5.0	7.0
l_{tr}	HSVM	49.4	186.0	93.0	98.9	5.0	7.5
	HSVM-S	**48.9**	**181.4**	**90.2**	**97.8**	**4.9**	**7.1**
$l_{\hat{uni}}$	HSVM	47.2	181.0	90.5	94.4	5.0	7.0
	HSVM-S	**46.9**	179.3	**88.7**	91.9	**4.9**	**6.9**
	HM3	70.1	172.1	-	88.8	5.2	7.4
$l_{\hat{sib}}$	HSVM	49.4	184.9	92.5	98.9	4.8	7.4
	HSVM-S	**48.9**	**170.2**	**91.6**	**90.8**	**4.7**	7.4
	HM3	64.8	172.9	-	92.7	4.8	7.1
$l_{\hat{sub}}$	HSVM	50.6	189.9	95.0	101.1	5.2	7.5
	HSVM-S	**47.2**	**169.4**	**85.2**	**89.4**	**4.3**	**6.6**
	HM3	65.0	170.9	-	91.9	4.8	7.2

number of nodes in the hierarchy is 188, with maximum depth 3.

We compared the performance of our proposed method HSVM−S with two algorithms: HSVM(Cai and Hofmann, 2007) and HM3(Rousu et al., 2006).

4.1 Effect of Different Loss Function

We compare the methods based on different loss functions, $l_{0/1}$, l_{Δ}, l_{tr}, $l_{\hat{u}ni}$, $l_{\hat{s}ib}$ and $l_{\hat{s}ub}$. The performances for three algorithms can be seen in Table 1. Those empty cells, denoted by "-", are not available in (Rousu et al., 2006).

As expected, $l_{0/1}$ is inferior to other hierarchical losses by getting poorest performance in all the testing losses, since it can not take into account the hierarchical information between categories. The results suggests that training with a hierarchical losses function, like $l_{\hat{s}ib}$ or $l_{\hat{u}ni}$, would lead to a better reduced $l_{0/1}$ on the test set as well as in terms of the hierarchical loss. In Table 1, we can also point out that when training with the same hierarchical loss, the performance of HSVM−S is better than HSVM under the measure of most hierarchical losses, since HSVM−S includes more hierarchical information,the relationship between the sibling categories, than HSVM which only separate the leave categories.

5 Conclusion

In this paper we present a hierarchical multi-class document categorization, which focus on maximize the margin of the classes at the different levels in the class hierarchy. In future work, we plan to extend the proposed hierarchical learning method to the case where the hierarchy is a DAG instead of tree and scale up the method further.

Acknowledgments

This work was (partially) funded by Chinese NSF 60673038, Doctoral Fund of Ministry of Education of China 200802460066, and Shanghai Science and Technology Development Funds 08511500302.

References

L. Cai and T Hofmann. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the ACM Conference on Information and Knowledge Management*.

L. Cai and T. Hofmann. 2007. Exploiting known taxonomies in learning overlapping concepts. In *Proceedings of International Joint Conferences on Artificial Intelligence*.

R. Caruana. 1997. Multi-task learning. *Machine Learning*, 28(1):41–75.

Ofer Dekel, Joseph Keshet, and Yoram Singer. 2004. Large margin hierarchical classification. In *Proceedings of the 21 st International Conference on Machine Learning*.

D. Koller and M Sahami. 1997. Hierarchically classifying documents using very few words. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. In *Journal of Machine Learning Research*.

A. Sun and E.-P Lim. 2001. Hierarchical text classification and evaluation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. In *Journal of Machine Learning*.

R. J. Vanderbei. 1999. Loqo: An interior point code for quadratic programming. In *Optimization Methods and Software*.

K. Wang, S. Zhou, and S Liew. 1999. Building hierarchical classifiers using class proximities. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*.

A. Weigend, E. Wiener, and J Pedersen. 1999. Exploiting hierarchy in text categorization. In *Information Retrieval*.

Toward finer-grained sentiment identification in product reviews through linguistic and ontological analyses

Hye-Jin Min
Computer Science Department
KAIST, Daejeon, KOREA
hjmin@nlp.kiast.ac.kr

Jong C. Park
Computer Science Department
KAIST, Daejeon, KOREA
park@nlp.kaist.ac.kr

Abstract

We propose categories of finer-grained polarity for a more effective aspect-based sentiment summary, and describe linguistic and ontological clues that may affect such fine-grained polarity. We argue that relevance for satisfaction, contrastive weight clues, and certain adverbials work to affect the polarity, as evidenced by the statistical analysis.

1 Introduction

Sentiment analysis have been widely conducted in several domains such as movie reviews, product reviews, news and blog reviews (Pang et al., 2002; Turney, 2002). The unit of the sentiment varies from a document level to a sentence level to a phrase-level, where a more fine-grained approach has been receiving more attention for its accuracy. Sentiment analysis on product reviews identifies or summarizes sentiment from reviews by extracting relevant opinions about certain attributes of products such as their parts, or properties (Hu and Liu, 2004; Popescu and Etzioni, 2005). Aspect-based sentiment analysis summarizes sentiments with diverse attributes, so that customers may have to look more closely into analyzed sentiments (Titov and McDonald, 2008). However, there are additional problems.

First, it is rather hard to choose the right level of detail. If concepts corresponding to attributes are too general, the level of detail may not be so much finer than the ones on a document level. On the other hand, if concepts are too specific, there may be some attributes that are hardly mentioned in the reviews, resulting in the data sparseness problem. Second, there are cases when some crucial information is lost. For example, suppose that two product attributes are mentioned in a sentence with a coordinated or subordinated structure. In this case, the information about their relation may not be shown in the summary if they are classified into different upper-level attributes. Consider (1).

(1) a. 옷은 맞지만/맞긴 한데, 색상이 너무 어두워요. osun macciman, sayksangi nemwu etwuweyo. 'It fits me okay, but the color is too dark.' (size: barely positive, color: negative)
b. 생각보다 좀 얇지만, 안에 받쳐 입는 거니까 나름 괜찮은거 같아요. sayngkakpota com yalpciman, aney patchye ipnun kenikka nalum kwaynchanhunke kathayo. 'It's a bit thinner than I thought, but it is good enough for layering.' (thickness: negative but acceptable, overall: positive)

Example (1) shows sample customer reviews about clothes, each first in Korean, followed by a Yale Romanized form, and an English translation. Note that the weight of the polarity in the sentiment about size e.g. in (1a) is overcome by the one about color. However, if the overall sentiment is computed by considering only the number of semantically identical phrases in the reviews, it misses the big picture.

In particular, when opinions regarding attributes are described with respect to expressions whose polarities are dependent on the specific contexts such as the weather or user preference, an overestimated or underestimated weight of the sentiment for each attribute may be assigned. In our example, 얇다/yalpta/'thin' has an ambiguous polarity, i.e., either positive or negative, whose real value depends on the expected utility of the clothes. In this case, the negative polarity is the intended one, as shown in (1b). In order to reflect this possibility, we need to adjust the weight of each polarity accordingly.

In this paper, we propose to look into the kind of linguistic and ontological clues that may in-

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 169–172,
Suntec, Singapore, 4 August 2009.

fluence the use of polarities, or the relevance for 'satisfaction of purchase' inspired by Kano's theory of quality element classification (Huiskonen and Pirttila, 1998), the conceptual granularities, and such syntactic and lexical clues as conjunction items and adverbs. They may play significant roles in putting together the identified polarity information, so as to assess correctly what the customers consider most important. We conducted several one-way Analysis of Variance (ANOVA) tests to identify the effects of each clue on deriving categories of polarity and quantification method 2 to see whether these clues can distinguish fine-grained polarities correctly.

Section 2 introduces categories of polarity. Section 3 analyzes ontological and linguistic clues for identifying the proper category. Section 4 describes our method to extract such clues for a statistical analysis. Section 5 discusses the results of the analysis and implications of the results. Section 6 concludes the paper.

2 Categories of polarity

We suggest two more fine-grained categories of polarity, or 'barely positive' (BP) and 'acceptably negative' (AN), in addition to positive (P), negative (N) and neutral (NEU). We distinguish 'barely positive' from normal positive and distinguish 'acceptably negative' from normal negative in order to derive finer-grained sentiments. Wilson and colleagues (2006) identified the strength of news articles in the MPQA corpus, where they separated intensity (low, medium, high) from categories (private states). For the purpose of identifying each attribute's contribution to the satisfaction after purchase, we believe that it is not necessary to have so many degrees of intensity. We argue that the polarity of 'barely positive' may hold attributes that must be satisfied and that 'acceptably negative' may hold those that are somewhat optional.

3 Linguistic and Ontological Analyses

In this section, we discuss linguistic and ontological clues that influence the process of identifying finer-grained polarity. For the purpose of exposition, we build hierarchical and aspect-based review structure as shown in Figure 1. Major aspects include Price, Delivery, Service, and Product. If we go down another level, Product is divided into Quality and Comfortableness. In defining relevant attributes, we consider all the lower-level concepts of major aspects, which contain the characteristics of the product with a description of the associated sentiment.

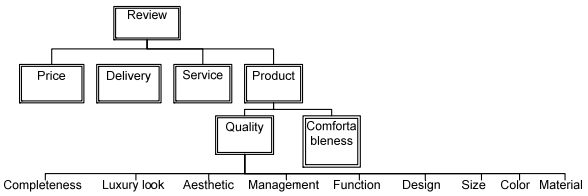


Figure 1. Review structure

Relevance for Satisfaction: We consider relevant attributes that affect the quality and satisfaction of the products as one of the important clues. Quality elements classified by Kano as shown in Table 1 can be base indicators of relevant attributes for satisfaction in real review text. For example, while completeness of the product may become crucial if the product has a defect, it is usually not the case that it would contribute much to the overall satisfaction of the customer.

Quality Elements	Example features
Must-be Quality (MQ)	Durability, Completeness
1-dimension Quality (1DQ)	Design, Color, Material
Attractive Quality (AQ)	Luxurious look

Table 1. Kano's Quality Elements

Conceptual Granularity: The concepts corresponding to attributes have a different level of detail. If the customer wants to comment on some attributes in detail, she could use a fine-grained concept (e.g., the width of the thigh part of the pants) rather than a coarse-grained one (e.g., just the size of the pants). To deal properly with the changing granularity of such concepts, we constructed a domain specific semi-hierarchical network for clothes of the ClothingType structure, in addition to the Review structure, by utilizing hierarchical category information in online shopping malls. Figure 2 shows an example for "pants".

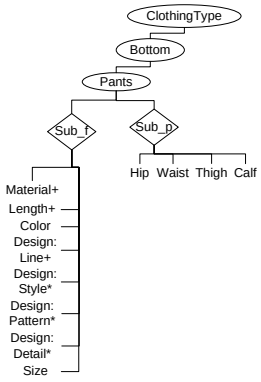


Figure 2. ClothingType structure for pants

Syntactic and Lexical Clues: Descriptions of each attribute in the reviews are often expressed

in a phrase or clause, so that conjunctions, or endings of a word with a conjunctive marker in Korean, play a significant role in connecting one attribute to another. They also convey a subtle meaning of the sentiment about relations between two or more connected attributes. We classified such syntactic clues into 4 groups of likeness (L), contrary (C), cause-effect (CE), and contrary with contrastive markers (CC).

Wilson and colleagues (2006) selected some syntactic clues as features for intensity classification. The selected features are shown to improve the accuracy, but the set of clues may vary to the nature of the given corpus, so that some otherwise useful clues that reflect a particular focused structure may not be selected. We argue that some syntactic clues such as the use of certain conjunctions can be identified manually to make up for the limitation of feature selection.

Adverbs modifying adjectives or verbs such as *too*, and *very* also strengthen the polarity of a given sentiment, so such clues work to differentiate normal positive or negative from 'barely positive' and 'acceptably negative'. Table 2 summarizes linguistic clues in the present analysis.

Clues	**Examples**	
CONJ/ END	L	-고 -ko 'and'
	C	-지만 -ciman 'but', 그러나 kulena 'however'
	CE	-어서 -ese 'so', 그래서 kulayse 'therefore'
	CC	-긴 –지만 -kin -ciman 'It's …, 'but', 'though'
ADV	Strong	매우 maywu 'very', 너무 nemwu 'too'
	Mild	좀 com 'a little'

Table 2. Syntactic and Lexical Clues

All these three types of clue that appear in the review text may interact with one another. For example, attributes with 'barely positive' tend to be described with a concept on a coarse level, and may belong to Must-be Quality (e,g., size in (1a)). However, if such attributes are negative, customers may explain them with a very fine-grained concept (e.g., the width of thigh is okay, but the calf part is too wide; interaction between relevance for satisfaction and conceptual granularity). They may also use adverbs such as 'too' to emphasize such unexpected polarity information. For emphasis, a contrastive structure can be used to indicate which attribute has a more weight (e.g., 'A but B'; interaction between syntactic clues and relevance for satisfaction). In addition, an unfocused attribute A may be the attribute with 'acceptably negative' if the polarity of the attribute B is positive. We believe that the interaction between lexical and syntactic clues and relevance for satisfaction are the most important and that this correlation information may be utilized with such fine-grained polarity as 'barely positive' or 'acceptably negative'.

4 Clue Acquisition

We acquired data semi-automatically for each clue from the extracted attributes and their descriptions from 500 product reviews of several types of pants and annotated polarities manually. We obtained raw text reviews from one of the major online shopping malls in Korea[1] and performed a morphology analysis and POS-tagging. After POS-tagging, we collected all the noun phrases as candidates of attributes. We regarded some of them as attributes with the following guidelines and filtered out the rest: 1) NP with frequent adjectives 2) NP with frequent non-functional and intransitive verbs. In the case of subject omission, we converted adjectives or verbs into their corresponding nouns, such as 'thin' into 'thickness'. Hu and Liu (2004) identified attributes of IT products based on frequent noun phrases and Popescu and Etzioni (2005) utilized PMI values between product class (hotels and scanners) and some phrases including product. In our case, we used attributes that belong only to the Product concept in the Review structure, because most attributes we consider are sub-types or sub-attribute of Product. The total number of <attribute, polarity> pairs is 474.

For relevance for satisfaction, we converted extracted attributes into one of the types of Kano's quality elements by the mapping table we built. For conceptual granularity we regarded all the attributes with a depth less than 2 as 'coarse' and those more than 2 as 'fine'. Syntactic and lexical clues are identified from the context information around extracted adjective or verbs by the patterns based on POS information.

5 Statistical Analysis and Discussion

We conducted one-way Analysis of Variance (ANOVA) tests using relevance for satisfaction (ReV), conceptual granularity (Granul), and two linguistic clues, ADV and CONJ/END, in order to assess the effects of each clue on identifying categories of polarity. The ANOVA suggests

[1] http://www.11st.co.kr

reliable effects of ReV ($F(2,474) = 22.2$; $p = .000$), ADV ($F(2, 474) = 41.3$; $p = .000$), and CONJ/END ($F(3, 474) = 6.1$; $p = .000$). We also performed post-hoc tests to test significant differences. For ReV, there are significant differences between 'MQ' and '1DQ' ($p=.000$), and between 'MQ' and 'AQ' ($p =.032$). AQ is related to 'positive' and MQ to 'acceptably negative' by the result. For ADV, there are significant differences between all pairs ($p <.05$). For CONJ/END, there are significant differences between 'likeness' and 'contrary' ($p = .015$), and between 'likeness' and 'contrary with contrastive markers' ($p = .025$). The 'contrary' and 'contrary with contrastive markers' types of conjunctions are related to 'acceptably negative'.

We also conducted Quantification method 2 to see if these clues can discriminate between BP and P and discriminate between AN and N. The regression equation for distinguishing AN from N is statistically significant at the 5% level ($F(7,177) = 12,2$; $R^2=0.335$; Std. error of the estimate = 0.821; error rate for discriminant = 0.21). The coefficients for 'mild' ($t^2=30.8$), 'contrary' ($t^2=17.8$) and 'contrary with contrastive markers' ($t^2=14.1$) are significant.

The results lead us to conclude that we can identify 'acceptably negative' from the clothes reviews by extracting the particular lexical clue, adverbs of 'mild' category and syntactic clue, such as conjunctions of 'contrary', and 'contrary with contrastive markers', or *contrastive weight*. This clue may convey the customer's argumentative intention toward the product, or *argumentative orientation*, for instance, A and B in '*A but B. C*' have different influence on the following discourse C (Elhadad and McKeown, 1990).

Although 'contrary with contrastive markers' plays an important role in identifying 'acceptably negative', it could also be used to identify another type of 'positive' as shown in example (2).

(2) 좀 두껍다는 생각이 듭니다. 그래도 따뜻하긴 하네요. com twukkeptanun sayngkaki tupnita. kulayto ttattushakin haneyyo. 'It is a bit thick, but it keeps me warm.'

It is a positive feature, but neither fully positive nor barely positive. It seems to be somewhere in-between. The order of appearance in reviews may also affect the strength of polarity. In addition, particular cue phrases such as ~것만 빼고/kesman ppayko/'except that …' can also convey 'acceptably negative', too.

In the future, we need to assess the importance of each proposed clue relative to others and to the existing ones. We also need to investigate the nature of interactions among linguistic, ontological and relevance for satisfaction clues, which may influence the actual performance for identifying finer-grained polarity.

6 Conclusion and Future Work

We proposed further categories of polarity in order to make aspect-based sentiment summary more effective. Our linguistic and ontological analyses suggest that there are clues, such as 'relevance for satisfaction', 'contrastive weight' and certain adverbials, that work to affect polarity in a more subtle but crucial manner, as evidenced also by the statistical analysis. We plan to find out product attributes that contribute most to modeling the interaction among the proposed clues in effective sentiment summarization.

Acknowledgments
This work was funded in part by the Intelligent Robotics Development Program, a 21st Century Frontier R&D Program by the Ministry of Knowledge Economy in Korea, and in part by the 2nd stage of the Brain Korea 21 project.

References

Ana-Maria Popescu and Oren Etzioni 2005. *Extracting Product Features and Opinions from Reviews*. Proc. HLT/EMNLP 2005, 339-346.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. *Thumbs up? Sentiment classification using machine learning techniques*. Proc. EMNLP.

Ivan Titov and Ryan McDonald 2008. *A Joint Model of Text and Aspect Ratings for Sentiment Summarization.* Proc. ACL-08: HLT, 308-316.

Janne Huiskonen and Timo Pirttila. 1998. *Sharpening logistic customer service strategy planning by applying Kano's quality element classification*. International Journal of Producion Economics, 56-57, 253-260, Elsevier Science B.V.

Michael Elhadad and Kathleen R. McKeown. 1990. *Generating Connectives*. Proc. COLING'97-101.

Minqing Hu and Bing Liu. 2004. *Mining and summarizing customer reviews*. Proc. ACM SIGKDD, 168–177. ACM Press.

Peter D. Turney. 2002. *Thumbs up or thumbs down? Sentiment orientation applied to unsupervised classification of reviews*. Proc. ACL, 417-424.

Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. 2006. *Recognizing Strong and Weak Opinion Clauses*. Computational Linguistics, 22 (2): 73-99.

Finding Hedges by Chasing Weasels: Hedge Detection Using Wikipedia Tags and Shallow Linguistic Features

Viola Ganter and **Michael Strube**
EML Research gGmbH
Heidelberg, Germany
http://www.eml-research.de/nlp

Abstract

We investigate the automatic detection of sentences containing linguistic hedges using corpus statistics and syntactic patterns. We take Wikipedia as an already annotated corpus using its tagged weasel words which mark sentences and phrases as non-factual. We evaluate the quality of Wikipedia as training data for hedge detection, as well as shallow linguistic features.

1 Introduction

While most research in natural language processing is dealing with identifying, extracting and classifying facts, recent years have seen a surge in research on sentiment and subjectivity (see Pang & Lee (2008) for an overview). However, even opinions have to be backed up by facts to be effective as arguments. Distinguishing facts from fiction requires to detect subtle variations in the use of linguistic devices such as linguistic hedges which indicate that speakers do not back up their opinions with facts (Lakoff, 1973; Hyland, 1998).

Many NLP applications could benefit from identifying linguistic hedges, e.g. question answering systems (Riloff et al., 2003), information extraction from biomedical documents (Medlock & Briscoe, 2007; Szarvas, 2008), and deception detection (Bachenko et al., 2008).

While NLP research on classifying linguistic hedges has been restricted to analysing biomedical documents, the above (incomplete) list of applications suggests that domain- and language-independent approaches for hedge detection need to be developed. We investigate Wikipedia as a source of training data for hedge classification. We adopt Wikipedia's notion of *weasel words* which we argue to be closely related to hedges and private states. Many Wikipedia articles contain a specific *weasel tag*, so that Wikipedia can be viewed as a readily annotated corpus. Based on this data, we have built a system to detect sentences that contain linguistic hedges. We compare a baseline relying on word frequency measures with one combining word frequency with shallow linguistic features.

2 Related Work

Research on hedge detection in NLP has been focused almost exclusively on the biomedical domain. Light et al. (2004) present a study on annotating hedges in biomedical documents. They show that the phenomenon can be annotated tentatively reliably by non-domain experts when using a two-way distinction. They also perform first experiments on automatic classification.

Medlock & Briscoe (2007) develop a weakly supervised system for hedge classification in a very narrow subdomain in the life sciences. They start with a small set of seed examples known to indicate hedging. Then they iterate and acquire more training seeds without much manual intervention (step 2 in their seed generation procedure indicates that there is some manual intervention). Their best system results in a 0.76 precision/recall break-even-point (BEP). While Medlock & Briscoe use words as features, Szarvas (2008) extends their work to n-grams. He also applies his method to (slightly) out of domain data and observes a considerable drop in performance.

3 Weasel Words

Wikipedia editors are advised to avoid *weasel words*, because they "offer an opinion without really backing it up, and ... are really used to express a non-neutral point of view."[1] Examples for weasel words as given by the style guide-

[1] http://en.wikipedia.org/wiki/Wikipedia:Guide_to_writing_better_articles

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 173–176,
Suntec, Singapore, 4 August 2009.

lines[2] are: *"Some people say ..."*, *"I think ..."*, *"Clearly ..."*, *"...is widely regarded as ..."*, *"It has been said/suggested/noticed ..."*, *"It may be that ..."* We argue that this notion is similar to linguistic hedging, which is defined by Hyland (1998) as "...any linguistic means used to indicate either a) a lack of complete commitment to the truth value of an accompanying proposition, or b) a desire not to express that commitment categorically." The Wikipedia style guidelines instruct editors to, if they notice weasel words, insert a {{`weasel-inline`}} or a {{`weasel-word`}} tag (both of which we will hereafter refer to as weasel tag) to mark sentences or phrases for improvement, e.g.

(1) `Others argue {{weasel-inline}} that the news media are simply catering to public demand.`

(2) `...therefore America is viewed by some {{weasel-inline}} technology planners as falling further behind Europe ...`

4 Data and Annotation

Weasel tags indicate that an article needs to be improved, i.e., they are intended to be removed after the objectionable sentence has been edited. This implies that weasel tags are short lived, very sparse and that – because weasels may not have been discovered yet – not all occurrences of linguistic hedges are tagged. Therefore we collected not one but several Wikipedia dumps[3] from the years 2006 to 2008. We extracted only those articles that contained the string {{`weasel`. Out of these articles, we extracted 168,923 unique sentences containing 437 weasel tags.

We use the dump completed on July 14, 2008 as development test data. Since weasel tags are very sparse, any measure of precision would have been overwhelmed by false positives. Thus we created a balanced test set. We chose one random, non-tagged sentence per tagged sentence, resulting (after removing corrupt data) in a set of 500 sentences. We removed formatting, comments and links to references from all dumps. As testing data we use the dump completed on March 6, 2009. It comprises 70,437 sentences taken from articles containing the string {{`weasel` with 328 weasel tags. Again, we created a balanced set of 500 sentences.

[2] `http://en.wikipedia.org/wiki/Wikipedia:Avoid_weasel_words`

[3] `http://download.wikipedia.org/`

	S	M	C
K	0.45	0.71	0.6
S		0.78	0.6
M			0.8

Table 1: Pairwise inter-annotator agreement

As the number of weasel tags is very low considering the number of sentences in the Wikipedia dumps, we still expected there to be a much higher number of potential weasel words which had not yet been tagged leading to false positives. Therefore, we also annotated a small sample manually. One of the authors, two linguists and one computer scientist annotated 100 sentences each, 50 of which were the same for all annotators to enable measuring agreement. The annotators labeled the data independently and following annotation guidelines which were mainly adopted from the Wikipedia style guide with only small adjustments to match our pre-processed data. We then used *Cohen's Kappa* (κ) to determine the level of agreement (Carletta, 1996). Table 4 shows the agreement between each possible pair of annotators. The overall inter-annotator agreement was $\kappa = 0.65$, which is similar to what Light et al. (2004) report but worse than Medlock & Briscoe's (2007) results. As Gold standard we merged all four annotations sets. From the 50 overlapping instances, we removed those where less than three annotators had agreed on one category, resulting in a set of 246 sentences for evaluation.

5 Method

5.1 Words Preceding Weasel Tags

We investigate the five words occurring right before each weasel tag in the corpus (but within the same sentence), assuming that weasel phrases contain at most five words and weasel tags are mostly inserted *behind* weasel words or phrases.

Each word within these 5-grams receives an individual score, based a) on the relative frequency of this word in weasel contexts and the corpus in general and b) on the average distance the word has to a weasel tag, if found in a weasel context. We assume that a word is an indicator for a weasel if it occurs close before a weasel tag. The final scoring function for each word in the training set

is thus:

$$Score(w) = RelF(w) + AvgDist(w) \quad (1)$$

with

$$RelF(w) = \frac{W(w)}{log_2(C(w))} \quad (2)$$

and

$$AvgDist(w) = \frac{W(w)}{\sum_{j=0}^{W(w)} dist(w, weaseltag_j)} \quad (3)$$

$W(w)$ denotes the number of times word w occurred in the context of a weasel tag, whereas $C(w)$ denotes the total number of times w occurred in the corpus. The basic idea of the $RelF$ score is to give those words a high score, which occur frequently in the context of a weasel tag. However, due to the sparseness of tagged instances, words that occur with a very high frequency in the corpus automatically receive a lower score than low-frequent words. We use the logarithmic function to diminish this effect.

In equation 3, for each weasel context j, $dist(w, weaseltag_j)$ denotes the distance of word w to the weasel tag in j. A word that always appears directly before the weasel tag will receive an $AvgDist$ value of 1, a word that always appears five words before the weasel tag will receive an $AvgDist$ value of $\frac{1}{5}$. The score for each word is stored in a list, based on which we derive the classifier (*words preceding weasel (wpw)*): Each sentence S is classified by

$$S \rightarrow weasel \;\; if \;\; wpw(S) > \sigma \quad (4)$$

where σ is an arbitrary threshold used to control the precision/recall balance and $wpw(S)$ is the sum of scores over all words in S, normalized by the hyperbolic tangent:

$$wpw(S) = \tanh \sum_{i=0}^{|S|} Score(w_i) \quad (5)$$

with $|S|$ = the number of words in the sentence.

5.2 Adding shallow linguistic features

A great number of the weasel words in Wikipedia can be divided into three categories:

1. Numerically underspecified subjects (*"Some people", "Experts", "Many"*)
2. Passive constructions (*"It is believed", "It is considered"*)
3. Adverbs (*"Often", "Probably"*)

We POS-tagged the test data with the TnT tagger (Brants, 2000) and developed finite state automata to detect such constellations. We combine these syntactic patterns with the word-scoring function from above. If a pattern is found, only the head of the pattern (i.e., adverbs, main verbs for passive patterns, nouns and quantifiers for numerically underspecified subjects) is assigned a score. The scoring function *adding syntactic patterns (asp)* for each sentence is:

$$asp(S) = \tanh \sum_{i=0}^{heads_S} Score(w_i) \quad (6)$$

where $heads_S$ = the number of pattern heads found in sentence S.

6 Results and Discussion

Both, the classifier based on *words preceding weasel (wpw)* and the one based on *added syntactic patterns (asp)* perform comparably well on the development test data. *wpw* reaches a 0.69 precision/recall break-even-point (BEP) with a threshold of σ = 0.99, while *asp* reaches a 0.70 BEP with a threshold of σ = 0.76.

Applied to the test data these thresholds yield an F-Score of 0.70 for *wpw* (prec. = 0.55/rec. = 0.98) and an F-score of 0.68 (prec. = 0.69/rec. = 0.68) for *asp* (Table 2 shows results at a few fixed thresholds allowing for a better comparison). This indicates that the syntactic patterns do not contribute to the regeneration of weasel tags. Word frequency and distance to the weasel tag are sufficient.

The decreasing precision of both approaches when trained on more tagged sentences (i.e., computed with a higher threshold) might be caused by the great number of unannotated weasel words. Indeed, an investigation of the sentences scored with the added syntactic patterns showed that many high-ranked sentences were weasels which had not been tagged. A disadvantage of the weasel tag is its short life span. The weasel tag marks a phrase that needs to be edited, thus, once a weasel word has been detected and tagged, it is likely to get removed soon. The number of tagged sentences is much smaller than the actual number of weasel words. This leads to a great number of false positives.

σ	.60	.70	**.76**	.80	.90	**.98**
balanced set						
wpw	.68	.68	.68	.69	.69	**.70**
asp	.67	.68	**.68**	.68	.61	.59
manual annot.						
wpw	-	.59	-	-	-	**.59**
asp	.68	.69	**.69**	.69	.70	.65

Table 2: F-scores at different thresholds (bold at the precision/recall break-even-points determined on the development data)

The difference between *wpw* and *asp* becomes more distinct when the manually annotated data form the test set. Here *asp* outperforms *wpw* by a large margin, though this is also due to the fact that *wpw* performs rather poorly. *asp* reaches an F-score of 0.69 (prec. = 0.61/rec. = 0.78), while *wpw* reaches only an F-Score of 0.59 (prec. = 0.42/ rec. = 1). This suggests that the added syntactic patterns indeed manage to detect weasels that have not yet been tagged.

When humans annotate the data they not only take specific words into account but the whole sentence, and this is why the syntactic patterns achieve better results when tested on those data. The word frequency measure derived from the weasel tags is not sufficient to cover this more intelligible notion of hedging. If one is to be restricted to words, it would be better to fall back to the weakly supervised approaches by Medlock & Briscoe (2007) and Szarvas (2008). These approaches could go beyond the original annotation and learn further hedging indicators. However, these approaches are, as argued by Szarvas (2008) quite domain-dependent, while our approach covers the entire Wikipedia and thus as many domains as are in Wikipedia.

7 Conclusions

We have described a hedge detection system based on word frequency measures and syntactic patterns. The main idea is to use Wikipedia as a readily annotated corpus by relying on its weasel tag. The experiments show that the syntactic patterns work better when using a broader notion of hedging tested on manual annotations. When evaluating on Wikipedia weasel tags itself, word frequency and distance to the tag is sufficient.

Our approach takes a much broader domain into account than previous work. It can also easily be applied to different languages as the weasel tag exists in more than 20 different language versions of Wikipedia. For a narrow domain, we suggest to start with our approach for deriving a seed set of hedging indicators and then to use a weakly supervised approach.

Though our classifiers were trained on data from multiple Wikipedia dumps, there were only a few hundred training instances available. The transient nature of the weasel tag suggests to use the Wikipedia edit history for future work, since the edits faithfully record all occurrences of weasel tags.

Acknowledgments. This work has been partially funded by the European Union under the project Judicial Management by Digital Libraries Semantics (JUMAS FP7-214306) and by the Klaus Tschira Foundation, Heidelberg, Germany.

References

Bachenko, Joan, Eileen Fitzpatrick & Michael Schonwetter (2008). Verification and implementation of language-based deception indicators in civil and criminal narratives. In *Proceedings of the 22nd International Conference on Computational Linguistics*, Manchester, U.K., 18–22 August 2008, pp. 41–48.

Brants, Thorsten (2000). TnT – A statistical Part-of-Speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing*, Seattle, Wash., 29 April – 4 May 2000, pp. 224–231.

Carletta, Jean (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

Hyland, Ken (1998). *Hedging in scientific research articles*. Amsterdam, The Netherlands: John Benjamins.

Lakoff, George (1973). Hedges: A study in meaning criteria and the logic of fuzzy concepts. *Journal of Philosophical Logic*, 2:458–508.

Light, Marc, Xin Ying Qiu & Padmini Srinivasan (2004). The language of Bioscience: Facts, speculations, and statements in between. In *Proceedings of the HLT-NAACL 2004 Workshop: Biolink 2004, Linking Biological Literature, Ontologies and Databases*, Boston, Mass., 6 May 2004, pp. 17–24.

Medlock, Ben & Ted Briscoe (2007). Weakly supervised learning for hedge classification in scientific literature. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic, 23–30 June 2007, pp. 992–999.

Pang, Bo & Lillian Lee (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Riloff, Ellen, Janyce Wiebe & Theresa Wilson (2003). Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the 7th Conference on Computational Natural Language Learning*, Edmonton, Alberta, Canada, 31 May – 1 June 2003, pp. 25–32.

Szarvas, György (2008). Hedge classification in biomedical texts with a weakly supervised selection of keywords. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Columbus, Ohio, 15–20 June 2008, pp. 281–289.

Mining User Reviews: from Specification to Summarization

Xinfan Meng
Key Laboratory of
Computational Linguistics
(Peking University)
Ministry of Education, China
mxf@pku.edu.cn

Houfeng Wang
Key Laboratory of
Computational Linguistics
(Peking University)
Ministry of Education, China
wanghf@pku.edu.cn

Abstract

This paper proposes a method to extract product features from user reviews and generate a review summary. This method only relies on product specifications, which usually are easy to obtain. Other resources like segmenter, POS tagger or parser are not required. At feature extraction stage, multiple specifications are clustered to extend the vocabulary of product features. Hierarchy structure information and unit of measurement information are mined from the specification to improve the accuracy of feature extraction. At summary generation stage, hierarchy information in specifications is used to provide a natural conceptual view of product features.

1 Introduction

Review mining and summarization aims to extract users' opinions towards specific products from reviews and provide an easy-to-understand summary of those opinions for potential buyers or manufacture companies. The task of mining reviews usually comprises two subtasks: product features extraction and summary generation.

Hu and Liu (2004a) use association mining methods to find frequent product features and use opinion words to predict infrequent product features. A.M. Popescu and O. Etzioni (2005) proposes OPINE, an unsupervised information extraction system, which is built on top of the KonwItAll Web information-extraction system. In order to reduce the features redundancy and provide a conceptual view of extracted features, G. Carenini et al. (2006a) enhances the earlier work of Hu and Liu (2004a) by mapping the extracted features into a hierarchy of features which describes the entity of interest. M. Gamon et al. (2005) clusters sentences in reviews, then label each cluster with a keyword and finally provide a tree map visualization for each product model. Qi Su et al. (2008) describes a system that clusters product features and opinion words simultaneously and iteratively.

2 Our Approach

To generate an accurate review summary for a specific product, product features must be identified accurately. Since product features are often domain-dependent, it is desirable that the features extraction system is as flexible as possible. Our approach are unsupervised and relies only on product specifications.

2.1 Specification Mining

Product specifications can usually be fetched from web sites like Amazon automatically. Those materials have several characteristics that are very helpful to review mining:

1. Nicely structured, provide a natural conceptual view of products;
2. Include only relevant information of the product and contain few noise words;
3. Except for the product feature itself, usually also provide a unit to measure this feature.

A typical mobile phone specification is partially given below:

- Physical features
  - Form: Mono block with full keyboard
  - Dimensions: 4.49 x 2.24 x 0.39 inch
  - Weight: 4.47 oz
- Display and 3D
  - Size: 2.36 inch
  - Resolution: 320 x 240 pixels (QVGA)

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 177–180,
Suntec, Singapore, 4 August 2009.

2.2 Architecture

The architecture of our approach. is depicted in Figure 1. We first retrieve multiple specifications from various sources like websites, user manuals etc. Then we run clustering algorithms on the specifications and generate a specification tree. And then we use this specification tree to extract features from product reviews. Finally the extracted features are presented in a tree form.

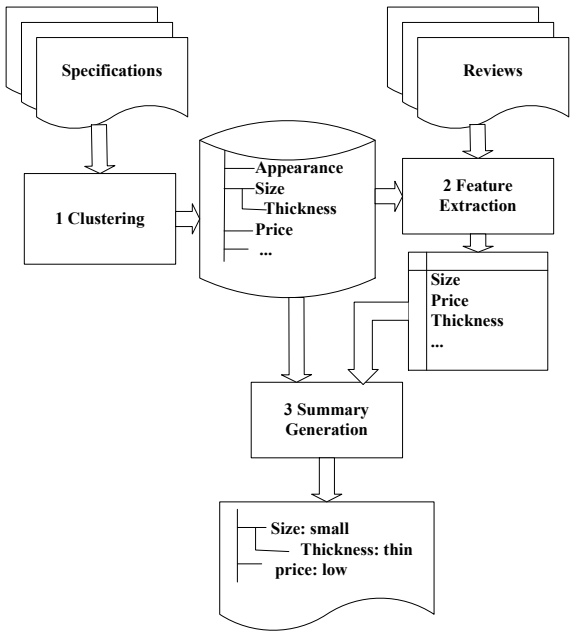


Figure 1: Architecture Overview

2.3 Specification Clustering

Usually, each product specification describes a particular product model. Some features are present in every product specification. But there are cases that some features are not available in all specifications. For instance, "WiFi" features are only available in a few mobile phones specifications. Also, different specifications might express the same features with different words or terms. So it is necessary to combine multiple specifications to include all possible features. Clustering algorithm can be used to combine specifications.

We propose an approach that takes following inherent information of specifications into account:

- **Hierarchy structure**: Positions of features in hierarchy reflect relationships between features. For example, "length", "width" feature are often placed under "size" feature.
- **Unit of measurement**: Similar features are usually measured in similar units. Though different specification might refer the same feature with different terms, the units of measurement used to describe those terms are usually the same. For example, "dimension" and "size" are different terms, but they share the same unit "mm" or "inch".

Naturally, a product can be viewed as a tree of features. The root is the product itself. Each node in the tree represents a feature in the product. A complex feature might be conceptually split into several simple features. In this case, the complex feature is represented as a parent and the simple features are represented as its children.

To construct such a product feature tree, we adopt the following algorithm:

- **Parse specifications**: We first build a dictionary for common units of measurement. Then for every specification, we use regular expression and unit dictionary to parse it to a tree of (feature, unit) pairs.
- **Cluster specification trees**: Given multiple specification trees, we cluster them into a single tree. Similarities between features are a combination of their lexical similarity, unit similarity and positions in hierarchy:

$$\begin{aligned} Sim(f1, f2) = & Sim_{lex}(f1, f2) \\ & + Sim_{unit}(f1, f2) \\ & + \alpha * Sim_{parent}(f1, f2) \\ & + (1 - \alpha) * Sim_{children}(f1, f2) \end{aligned}$$

  The parameter α is set to 0.7 empirically. If $Sim(f1, f2)$ is larger than 5, we merge features $f1$ and $f2$ together.

After clustering, we can get a specification tree resembles the one in subsection 2.1. However, this specification tree contains much more features than any single specification.

2.4 Features Extraction

Features described in reviews can be classified into two categories: explicit features and implicit features (Hu and Liu, 2004a). In the following sections, we describe methods to extract features in Chinese product reviews. However, these methods are designed to be flexible so that they can be easily adapted to other languages.

2.4.1 Explicit Feature Extraction

We generate bi-grams in character level for every feature in the specification tree, and then match them to every sentence in the reviews. There might be cases that some bi-grams would overlap or concatenated. In these cases, we join those bi-grams together to form a longer expression.

2.4.2 Implicit Feature Extraction

Some features are not mentioned directly but can be inferred from the text. Qi Su et al. (2008) investigates the problem of extracting those kinds of features. There approach utilizes the association between features and opinion words to find implicit features when opinion words are present in the text. Our methods consider another kind of association: the association between features and units of measurement. For example, in the sentence "A mobile phone with 8 mega-pixel, not very common in the market." feature name is absent in the sentence, but the unit of measurement "mega pixel" indicates that this sentence is describing the feature "camera resolution".

We use regular expression and dictionary of unit to extract those features.

2.5 Summary Generation

There are many ways to provide a summary. Hu and Liu (2004b) count the number of positive and negative review items towards individual feature and present these statistics to users. G. Carenini et al. (2006b) and M. Gamon et al. (2005) both adopt a tree map visualization to display features and sentiments associated with features.

We adopt a relatively simple method to generate a summary. We do not predict the polarities of the user's overall attitudes towards product features. Predicting polarities might entail the construction of a sentiment dictionary, which is domain dependent. Also, we believe that text descriptions of features are more helpful to users. For example, for feature "size", descriptions like "small" and "thin" are more readable than "positive".

Usually, the words used to describe a product feature are short. For each product feature, we report several most frequently occurring uni-grams and bi-grams as the summary of this feature. In Figure 2, we present a snippet of a sample summary output.

- mobile phone: not bad, expensive
  - appearance: cool
    - color: white
    - size: small, thin
  - camera functionality: so-so, acceptable
    - picture quality: good
    - picture resolution: not high
  - entertainment functionality: powerful
    - game: fun, simple

Figure 2: A Summary Snippet

3 Experiments

In this paper, we mainly focus on Chinese product reviews. The experimental data are retrieved from ZOL websites (`www.zol.com.cn`). We collected user reviews on 2 mobile phones, 1 digital camera and 2 notebook computers. To evaluate performance of our algorithm on real-world data, we do not perform noise word filtering on these data. Then we have a human tagger to tag features in the user reviews. Both explicit features and implicit features are tagged.

No. of Clustering Specifications	Mobile Phone	Digital Camera	Notebook Computer
1	153	101	102
5	436	312	211
10	520	508	312

Table 1: No. of Features in Specification Trees.

The specifications for all 3 kinds of products are retrieved from ZOL, PConline and IT168 websites. We run the clustering algorithm on the specifications and generate a specification tree for each kind of product. Table 1 shows that our clustering method is effective in collecting product features. The number of features increases rapidly with the number of specifications input into clustering algorithm. When we use 10 specifications as input, the clustering methods can collect several hundred features.

Then we run our algorithm on the data and evaluate the precision and recall. We also run the algorithms described in Hu and Liu (2004a) on the same data as the baseline.

From Table 2, we can see the precision of baseline system is much lower than its recall. Examining the features extracted by baseline system, we find that many mistakenly recognized features are high-frequency words. Some of those words appear many times in text. They are related to prod-

Product Model	No. of Features	Hu and Liu's Approach			the Proposed Approach		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Mobile Phone 1	507	0.58	0.74	0.65	0.69	0.78	0.73
Mobile Phone 2	477	0.59	0.65	0.62	0.71	0.77	0.74
Digital camera	86	0.56	0.68	0.61	0.69	0.78	0.73
Notebook Computer 1	139	0.41	0.63	0.50	0.70	0.74	0.72
Notebook Computer 2	95	0.71	0.88	0.79	0.76	0.88	0.82

Table 2: Precision and Recall of Product Extraction.

uct but are not considered to be features. Some examples of these words are "advantages", "disadvantages" and "good points" etc. And many other high-frequency words are completely irrelevant to product reviews. Those words include "user", "review" and "comment" etc. In contrast, our approach recognizes features by matching bigrams to the specification tree. Because those high-frequency words usually are not present in specifications. They are ignored by our approach. Thus from Table 2, we can conclude that our approach could achieve a relatively high precision while keep a high recall.

Product Model	Precision
Mobile Phone 1	0.78
Mobile Phone 2	0.72
Digital camera	0.81
Notebook Computer 1	0.73
Notebook Computer 2	0.74

Table 3: Precision of Summary.

After the summary is given, for each word in summary, we ask one person to decide whether this word correctly describe the feature. Table 3 gives the summary precision for each product model. In general, on-line reviews have several characteristics in common. The sentences are usually short. Also, words describing features usually co-occur with features in the same sentence. Thus, when the features in a sentence are correctly recognized, Words describing those features are likely to be identified by our methods.

4 Conclusion

In this paper, we describe a simple but effective way to extract product features from user reviews and provide an easy-to-understand summary. The proposed approach is based only on product specifications. The experimental results indicate that our approach is promising.

In future works, we will try to introduce other resources and tools into our system. We will also explore different ways of presenting and visualizing the summary to improve user experience.

Acknowledgments

This research is supported by National Natural Science Foundation of Chinese (No.60675035) and Beijing Natural Science Foundation (No.4072012).

References

M. Hu and B. Liu. 2004a. *Mining and Summarizing Customer Reviews*. In Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining, pages 168-177. ACM Press New York, NY, USA.

M. Hu and B. Liu. 2004b. *Mining Opinion Features in Customer Reviews*. In Proceedings of Nineteenth National Conference on Artificial Intelligence.

M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger. 2005. *Pulse: Mining Customer Opinions from Free Text*. In Proceedings of the 6th International Symposium on Intelligent Data Analysis.

A.M. Popescu and O. Etzioni. 2005. *Extracting Product Features and Opinions from reviews*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP).

Giuseppe Carenini, Raymond T. Ng, and Adam Pauls. 2006a. *Multi-Document Summarization of Evaluative Text*. In Proceedings of the conference of the European Chapter of the Association for Computational Linguistics.

Giuseppe Carenini, Raymond T. Ng, and Adam Pauls. 2006b. *Interactive multimedia summaries of evaluative text*. In Proceedings of Intelligent User Interfaces (IUI), pages 124-131. ACM Press, 2006.

Qi Su, Xinying Xu, Honglei Guo, Zhili Guo, Xian Wu, Xiaoxun Zhang, Bin Swen. 2008. *Hidden Sentiment Association In Chinese Web Opinion Mining*. In Proceedings of the 17th International Conference on the World Wide Web, pages 959-968.

An Ontology–Based Approach for Key Phrase Extraction

Chau Q. Nguyen
HCM University of Industry
12 Nguyen Van Bao St, Go Vap Dist,
HCMC, Vietnam
chauqn@hui.edu.vn

Tuoi T. Phan
HCMC University of Technology
268 Ly Thuong Kiet St, Dist 10,
HCMC, Vietnam
tuoi@cse.hcmut.edu.vn

Abstract

Automatic key phrase extraction is fundamental to the success of many recent digital library applications and semantic information retrieval techniques and a difficult and essential problem in Vietnamese natural language processing (NLP). In this work, we propose a novel method for key phrase extracting of Vietnamese text that exploits the Vietnamese Wikipedia as an ontology and exploits specific characteristics of the Vietnamese language for the key phrase selection stage. We also explore NLP techniques that we propose for the analysis of Vietnamese texts, focusing on the advanced candidate phrases recognition phase as well as part-of-speech (POS) tagging. Finally, we review the results of several experiments that have examined the impacts of strategies chosen for Vietnamese key phrase extracting.

1 Introduction

Key phrases, which can be single keywords or multiword key terms, are linguistic descriptors of documents. They are often sufficiently informative to allow human readers get a feel for the essential topics and main content included in the source documents. Key phrases have also been used as features in many text-related applications such as text clustering, document similarity analysis, and document summarization. Manually extracting key phrases from a number of documents is quite expensive. Automatic key phrase extraction is a maturing technology that can serve as an efficient and practical alternative.

In this paper, we present an ontology-based approach to building a Vietnamese key phrase extraction system for Vietnamese text. The rest of the paper is organized as follows: Section 2 states the problem as well as describes its scope, Section 3 introduces resources of information in Wikipedia that are essential for our method, Section 4 describes extraction of titles and its categories from Wikipedia to build a dictionary, Section 5 proposes a methodology for the Vietnamese key phrase extraction model, Section 6 evaluates our approach on many Vietnamese query sentences with different styles of texts, and finally the conclusion is presented in Section 7.

2 Background

The objective of our research is to build a system that can extract key phrases in Vietnamese queries in order to meet the demands associated with information searching and information retrieving, especially to support search engines and automatic answer systems on the Internet. For this purpose, we provide the following definition:

Key phrases in a sentence are phrases that express meaning completely and also express the purpose of the sentence to which they are assigned.

For an example, we have a query sentence as follows: "*Laptop Dell E1405 có giá bao nhiêu?*". That means "*How much does a Dell E1405 laptop cost?* ".

Key phrases are "*Laptop Dell E1405*", "*giá*", and "*bao nhiêu*". In this case, the interrogative word "*bao nhiêu*" is used to add a meaning for the two rest noun phrases, making the query of users clear, wanting to know the numeral aspect about the "*price*" of a "*Laptop Dell E1405*".

3 Wikipedia

Wikipedia is a multilingual, web-based, freely available encyclopedia, constructed as a collaborative effort of voluntary contributors on the web. Wikipedia grows rapidly, and with approximately 7.5 million articles in more than 253 languages, it has arguably become the world's largest collection of freely available knowledge.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 181–184,
Suntec, Singapore, 4 August 2009.

Wikipedia contains a rich body of lexical semantic information, the aspects of which are comprehensively described in (Zesch et al., 2007). Additionally, the redirect system of Wikipedia articles can be used as a dictionary for synonyms, spelling variations and abbreviations.

A PAGE. A basic entry in Wikipedia is a page that represents either a normal Wikipedia article, a redirect to an article, or a disambiguation page. Each page object provides access to the article text (with markup information or as plain text), the assigned categories, the ingoing and outgoing article links as well as all redirects that link to the article.

A LINK. Each page consists of many links which function not only to point from the page to others, but also to guide readers to pages that provide additional information about the entries mentioned. Each link is associated with an anchor text that denotes an ambiguous name or is an alternative name, instead of a canonical name.

CATEGORY. Category objects represent Wikipedia categories and allow access to the articles within each category. As categories in Wikipedia form a thesaurus, a category object also provides means to retrieve parent and child categories as well as siblings and all recursively collected descendants.

REDIRECT PAGE. A redirect page typically contains only a reference to an entry or a concept page. The title of the redirect page is an alternative name for that entity or concept.

DISAMBIGUATION PAGE. A disambiguation page is created for an ambiguous name that denotes two or more entities in Wikipedia. It consists of links to pages that define different entities with the same name.

4 Building a dictionary

Based on the aforementioned resources of information, we follow the method presented in (Bunescu and Pasca, 2006) to build a dictionary called ViDic. Since our research focuses on Key phrases, we first consider which pages in Wikipedia define concepts or objects to which key phrases refer. The key phrases are extracted from the title of the page. We consider a page has key phrases if it satisfies one of the following steps:

1. If its title is a word or a phrase then the title is key phrase.
2. If its title is a sentence then we follow the method presented in (Chau and Tuoi, 2007) to extract key phrases of the sentence.

Following this method, the ViDic is constructed so that the set of entries in the ViDic consists of all strings that denote a concept. In particular, if c is a concept, its key phrases, its title name, its redirect name and its category are all added as entries in the ViDic. Then each entry string in the ViDic is mapped to a set of entries that the string may denote in Wikipedia. As a result, a concept c is included in the set if, and only if, the string has key phrases which is extracted from the title name, redirect name, or disambiguation name of c.

Although we utilize information from Wikipedia to build the ViDic, our method can be adapted for an ontology or knowledge base in general.

5 Proposed method

We consider the employment of a set of NLP techniques adequate for dealing with the Vietnamese key phrase extraction problem. We propose the following general Vietnamese key phrase extraction model (see Figure 1).

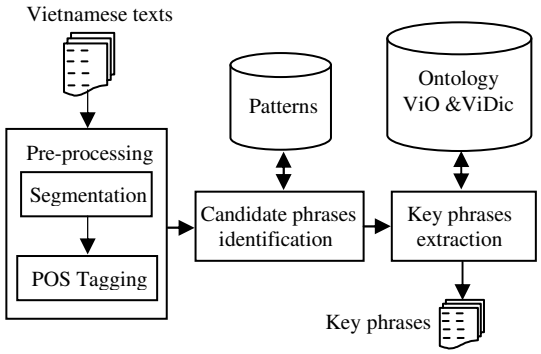


Figure 1. The general Vietnamese key phrase extraction model.

5.1 Pre-processing

The input of pre-processing is user's queries and the output is a list of words and their POS labels. Because of the effectiveness and convenience associated with integrating two stages of word segmentation and POS tagging, we proposed two modules for the pre-processing stage. The purposes of two modules are as follows:

- **Word Segmentation**: The main function of this segmentation module is to identify and separate the tokens present in the text in such a way that every individual word, as well as every punctuation mark, will be a different token. The segmentation module considers words, numbers with decimals or dates in nu-

merical format in order not to separate the dot, the comma or the slash (respectively) from the preceding and/or following elements.

- **POS tagging**: The output of the segmentation module is taken as input by the POS tagging module. Almost any kind of POS tagging could be applied. In our system, we have proposed a hybrid model for the problem of Vietnamese POS Tagging (Chau and Tuoi, 2006). This model combines a rule-based method and a statistical learning method. With regard to data, we use a lexicon with information about possible POS tags for each word, a manually labeled corpus, syntax and context of texts.

5.2 Candidate phrases identification

The input of the candidate phrase identification is a list of words and their POS labels, and the output is a list of words and their chunking labels. The idea underlying this method (Chau and Tuoi, 2007) for the Vietnamese key phrase extraction is based on a number of grammatical constructions in Vietnamese. The method consists of pattern-action rules executed by the finite-state transduction mechanism. It recognizes entities such as noun phrases. In order to accomplish the noun phrases recognition, we have developed over 434 patterns of noun phrase groups that cover proper noun constructs.

5.3 Key phrases extraction

In this section, we focus on the description of a methodology for key phrase extraction. This method combines a pattern-based method and a statistical learning method. Both methods will complement each other to increase the expected performance of the model. In particular, the method has the following steps:

• Step 1: We propose a method that exploits specific characteristics of Vietnamese (Chau and Tuoi, 2007). At the heart of this method is the idea of building a Vietnamese words set that reflects semantic relationships among objects. For example, consider the sentence that follows:

"Máy tính này có dung lượng RAM lớn nhất là bao nhiêu ?" that means "*What is the largest RAM capacity for this computer?"*

In this sentence, we have two objects "*Máy tính"(this computer)* and "*RAM"* in real world. Respectively, two noun phrases are *"Máy tính"(this computer)* and "*dung lượng RAM lớn nhất" (the largest RAM capacity).* We consider the meanings of words per the above example; we will recognize "*có*", a meaning word in our meaning word set, which reflects a possessive relationship between *"Máy tính"* and "*dung lượng RAM lớn nhất"*. This has identified "*dung lượng RAM lớn nhất"* representing the meaning of the sentence.

This meaning word-based approach provides a set of semantic relationships (meaning words) between phrases to support key phrase extraction, which does not require building a hierarchy or semantic network of objects in the Vietnamese language.

• Step 2: In case the sentence has no meaning word among phrases, the key phrase extracting process is based on the ViO ontology via concept matching. In particular, this step has the following phases:

1. every candidate phrase in the sentence is matched to an entry in the VicDic dictionary especially when new phrases are not a concern or do not exist in the dictionary. Because a partial matching dilemma usually exists, we apply several strategies to improve the matching process, including maximum matching, minimum-matching, forward-matching, backward-matching and bi-directional matching.
2. if the matching process is successful, then we retrieve categories for the entries respectively via the category system in the ViO ontology; if the candidate phrase has the most specific category, then the phrase is the key phrase of the sentence indicated in Step 3.
3. if the matching process is not successful, then we find a semantic similarity concept in the ViO ontology as Step 4. After that, the key phrase extracting process will go to phase 2.

• Step 3: The idea of the most specific category identification process based on the ViO ontology is shown as pseudo-code, such as

Algorithm: the most specific category identification

- Input: C_1, C_2 categories, and the ViO Ontology
- Output: C_1 or C_2 or both C_1 and C_2

1. **begin**
2. **if** C_1& C_2 have a synonyms relationship in ViO
3. **then** C_1 & C_2 are the most specific categories
4. **else if** C_1 has isa relationship of C_2 **then** C_1 is the most specific category.
5. to traverse the ViO ontology from C_1 & C_2 to find the nearest common ancestor node (C'). Calculate the distance between C_1 and C' (h_1), distance C_2 and C' (h_2).
6. **if** $h_1 > h_2$ **then** C_1 is the most specific category
7. **else if** $h_1 < h_2$ **then** C_2 is the most specific

category
8. **else** C_1 & C_2 are the most specific categories
9. **end;**

• Step 4: To find the semantic similarity concept for each concept *t* that is still unknown after phase 2, we traverse the ontology hierarchy from its root to find the best node. We choose the semantic similarity that was described as in (Banerjee and Pederson, 2003). However, we do not use the whole formula. In particular , we use a similar formula that is specified as follows:

$$Acu\_Sim(w, c) = Sim(w, c) + \sum Sim(w, c')$$

in which, w is the phrase that needs to be annotated, c is the candidate concept and c' is the concept that is related to c.

At the current node c while traversing, the similarity values between t and all children of c are calculated. If the maximum of similarity values is less than similarity value between t and c, then c is the best node corresponding to t. Otherwise, continue the procedure with the current node as the child node with the maximum similarity value. The procedure stops when the best node is found or it reaches a leaf node.

6 Evaluation

To evaluate the result of the proposed model, we use **recall** and **precision** measures that are defined as in (Chau & Tuoi, 2007). In order to test the model we selected a questions set from sources on the web as follows:

- TREC (Text REtrieval Conference) (http://trec.nist.gov/data/): TREC-07 (consisting of 446 questions); TREC-06 (consisting of 492 questions); and TREC-02 (consisting of 440 questions).
- The web page www.lexxe.com: consisting of 701 questions.

After that, the question set (consisting of 2079 questions) is translated into a Vietnamese questions set, we called D_1 dataset. All key phrases of the D_1 dataset are manually extracted by two linguists for the quality of the dataset. Then we have two versions respectively, V_1 and V_2. The results of our system is shown as follows:

Ver	R	A	Ra	Precision	Recall
V_1	3236	3072	2293	74.6%	70.8%
V_2	3236	3301	2899	**89.6%**	**87.8%**

Table 1. Results of Vietnamese key phrase extraction.

7 Conclusion

We have proposed an original approach to key phrase extraction. It is a hybrid and incremental process for information searching for search engines and automatic answer systems in Vietnamese. We achieved precision of around 89.6% for our system. The experimental results have show that our method achieves high accuracy.

Currently, Wikipedia editions are available for approximately 253 languages, which means that our method can be used to build key phrase systems for a large number of languages. In spite of the exploitation of Wikipedia as a Vietnamese ontology, our method can be adapted for any ontology and knowledge base in general.

Furthermore, we had to construct all necessary linguistic resources and define all data structures from scratch, while enjoying some advantages derived from the many existent methodologies for morpho-syntactic annotation and the high consciousness of a standardization tendency. Specifically, we built a set with 434 noun phrase patterns and a rules set for Vietnamese key phrase identification. Our patterns and rules set can be easily readjusted and extended. The results obtained lay the foundation for further research in NLP for Vietnamese including text summarization, information retrieval, information extraction, etc.

References

Bunescu, R., Pasca, M. 2006. Using encyclopedic knowledge for name entity disambiguation. In *Proceedings of the 11th Conference of EACL*:9-16.

Banerjee S.,Pederson T., 2003. Extended Gloss Overlaps as a Measure of Semantic Relatedness, In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*: 805–810.

Chau Q.Nguyen, Tuoi T.Phan. 2007. A Pattern-based Approach to Vietnamese Key Phrase Extraction, In *Addendum Contributions of the 5th International IEEE Conference on Computer Sciences- RIVF'07*: 41-46.

Chau Q.Nguyen, Tuoi T.Phan. 2006. A Hybrid Approach to Vietnamese Part-Of-Speech Tagging. In *Proceedings of the 9th International Oriental COCOSDA Conference (O-COCOSDA'06)*, Malaysia:157-160.

Zesch, T., Gurevych, I. 2007. Analysis of the Wikipedia Category Graph for NLP Applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT 2007)*:1–8.

Query Segmentation Based on Eigenspace Similarity

Chao Zhang [†] [‡] **Nan Sun** [‡] **Xia Hu** [‡] **Tingzhu Huang** [†] **Tat-Seng Chua** [‡]

[†]School of Applied Math
University of Electronic Science
and Technology of China,
Chengdu, 610054, P.R. China
zhangcha@comp.nus.edu.sg
tzhuang@uestc.edu.cn

[‡]School of Computing
National University of Singapore,
Computing 1, Singapore 117590
{sunn,huxia,chuats}@comp.nus.edu.sg

Abstract

Query segmentation is essential to query processing. It aims to tokenize query words into several semantic segments and help the search engine to improve the precision of retrieval. In this paper, we present a novel unsupervised learning approach to query segmentation based on principal eigenspace similarity of query-word-frequency matrix derived from web statistics. Experimental results show that our approach could achieve superior performance of 35.8% and 17.7% in F-measure over the two baselines respectively, i.e. MI (Mutual Information) approach and EM optimization approach.

1 Introduction

People submit concise word-sequences to search engines in order to obtain satisfying feedback. However, the word sequences are generally ambiguous and often fail to convey the exact information to search engine, thus severely, affecting the performance of the system. For example, given the query "free software testing tools download". A simple bag-of-words query model cannot analyze "software testing tools" accurately. Instead, it returns "free software" or "free download" which are high frequency web phrases. Therefore, how to segment a query into meaningful semantic components for implicit description of user's intention is an important issue both in natural language processing and information retrieval fields.

There are few related studies on query segmentation in spite of its importance and applicability in many query analysis tasks such as query suggestion, query substitution, etc. To our knowledge, three approaches have been studied in previous works: MI (Mutual Information) approach (Jones et al., 2006; Risvik et al., 2003), supervised learning approach (Bergsma and Wang, 2007) and EM optimization approach (Tan and Peng, 2008). However, MI approach calculates MI value just between two adjacent words that cannot handle long entities. Supervised learning approach requires a sufficiently large number of labeled training data, which is not conducive in real applications. EM algorithm often converges to a local maximum that depends on the initial conditions. There are also many relevant research on Chinese word segmentation (Teahan et al., 2000; Peng and Schuurmans, 2001; Xu et al., 2008). However, they cannot be applied directly to query segmentation (Tan and Peng, 2008).

Under this scenario, we propose a novel unsupervised approach for query segmentation. Differing from previous work, we first adopt the n-gram model to estimate the query term's frequency matrix based on word occurrence statistics on the web. We then devise a new strategy to select principal eigenvectors of the matrix. Finally we calculate the similarity of query words for segmentation. Experimental results demonstrate the effectiveness of our approach as compared to two baselines.

2 Methodology

In this Section, we introduce our proposed query segmentation approach, which is based on query word frequency matrix principal eigenspace similarity. To facilitate understanding, we first present a general overview of our approach in Section 2.1 and then describe the details in Section 2.2-2.5.

2.1 Overview

Figure 1 briefly shows the main procedure of our proposed query segmentation approach. It starts with a query which consists of a vector of words$\{w_1 w_2 \cdots w_n\}$. Our approach first build a query-word frequency matrix M based on web statistics to describe the relationship between any

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 185–188,
Suntec, Singapore, 4 August 2009.

two query words (Step 1). After decomposing M (step 2), the parameter k which defines the number of segments in the query is estimate in Step 3. Besides, a principal eigenspace of M is built and the projection vectors($\{\alpha_i\}, i \in [1, n]$) associated with each query-word are obtained (Step 4). Similarities between projection vectors are then calculated, which determine whether the corresponding two words should be segmented together (Step5). If the number of segmented components is not equal to k, our approach modifies the threshold δ and repeats steps 5 and 6 until the correct k number of segmentations are obtained(Step 7).

Input:	one n words query: $w_1 w_2 \cdots w_n$;
Output:	k segmented components of query;
Step 1:	Build a frequency matrix M (Section 2.2);
Step 2:	Decompose M into sorted eigenvalues and eigenvectors;
Step 3:	Estimate parameter k (Section 2.4);
Step 4:	Build principal eigenspace with first k eigenvectors and get the projection ($\{\alpha_i\}$) of M in principal eigenspace (Section 2.3);
Step 5:	Segment the query: if $(\alpha_i \cdot \alpha_j^T)/(\|\alpha_i\| \cdot \|\alpha_j\|) \geq \delta$, segment w_i and w_j together (Section 2.5)
Step 6:	If the number of segmented parts does not equal to k, modify δ, go to step 5;
Step 7:	output the right segmentations

Figure 1: Query Segmentation based on query-word-frequency matrix eigenspace similarity

2.2 Frequency Matrix

Let $W = w_1, w_2, \cdots, w_n$ be a query of n words. We can build the relationships of any two words using a symmetric matrix: $M = \{m_{i,j}\}_{n \times n}$

$$m_{i,j} = \begin{cases} F(w_i) & \text{if } i = j \\ F(w_i w_{i+1} \cdots w_j) & \text{if } i < j \\ m_{j,i} & \text{if } i > j \end{cases} \quad (1)$$

$$F(w_i w_{i+1} \cdots w_j) = \frac{count(w_i w_{i+1} \cdots w_j)}{\sum_{i=1}^{n} w_i} \quad (2)$$

Here $m_{i,j}$ denotes the correlation between $(w_i \cdots w_{j-1})$ and w_j, where $(w_i \cdots w_{j-1})$ means a sequence and w_j is a word. Considering the difference of each matrix element $m_{i,j}$, we normalize $m_{i,j}$ with:

$$m_{i,j} = 2 \cdot m_{i,j}/(m_{i,i} + m_{j,j}) \quad (3)$$

$F(\cdot)$ is a function measuring the frequency of query words or sequences. To improve the precision of measurement and reduce the computation cost, we adopt the approach proposed by (Wang et al., 2007) here. First, we extract the relevant documents associated with the query via Google Soap Search API. Second, we count the number of all possible n-gram sequences which are highlighted in the titles and snippets of the returned documents. Finally, we use Eqn.(2) to estimate the value of $m_{i,j}$.

2.3 Principal Eigenspace

Although matrix M depicts the correlation of query words, it is rough and noisy. Under this scenario, we transform M into its principal eigenspace which is spanned by k largest eigenvectors, and each query word is denoted by the corresponding eigenvector in the principal eigenspace.

Since M is a symmetric positive definite matrix, its eigenvalues are real numbers and the corresponding eigenvectors are non-zero and orthotropic to each other. Here, we denote the eigenvalues of M as : $\lambda(M) = \{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. All eigenvalues of M have corresponding eigenvectors:$V(M) = \{x_1, x_2, \cdots, x_n\}$.

Suppose that principal eigenspace $\overline{M}(\overline{M} \in \mathcal{R}^{n \times k})$ is spanned by the first k eigenvectors, i.e. $\overline{M} = Span\{x_1, x_2, \cdots x_k\}$, then row i of M can be represented by vector α_i which denotes the i-th word for similarity calculation in Section 2.5, and α_i is derived from:

$$\{\alpha_1^T, \alpha_2^T, \cdots, \alpha_n^T\}^T = \{x_1, x_2, \cdots, x_k\} \quad (4)$$

Section 2.4 discusses the details of how to select the parameter k.

2.4 Parameter k Selection

PCA (principal component analysis) (Jolliffe, 2002) often selects k principal components by the following criterion:

k is the smallest integer which satisfies:

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq Threshold \quad (5)$$

where n is the number of eigenvalues. When $\lambda_k \gg \lambda_{k+1}$, Eqn.(5) is very effective. However, according to the Gerschgorin circle theorem, the non-diagonal values of M are so small that the eigenvalues cannot be distinguished easily. Under this circumstance, a prefixed threshold is too restrictive to be applied in complex situations. Therefore a function of n is introduced into the threshold as follows:

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq (\frac{n-1}{n})^2 \quad (6)$$

If k eigenvalues are qualified to be the principal components, then the threshold in Eqn.(5) cannot be lower than 0.5, and need not be higher than $\frac{n-1}{n}$. If the length of the shortest query we segmented is 4, we choose $(\frac{n-1}{n})^2$ because it will be smaller than $\frac{n-1}{n}$ and larger than 0.5 with n no smaller than 4.

The k eigenvectors will be used to segment the query into k meaningful segments (Weiss, 1999; Ng et al., 2001). In the k-dimensional principal eigenspace, each dimension of the space describes a semantic concept of the query. When one eigenvalue is bigger, the corresponding dimension contains more query words.

2.5 Similarity Computation

If the word i and word j are co-occurrence, α_i and α_j are approximately parallel in the principal eigenspace; otherwise, they are approximately orthogonal to each other. Hence, we measure the similarity of α_i and α_j with inner-product to perform the segmentation (Weiss, 1999; Ng et al., 2001). Selecting a proper threshold δ, we segment the query using Eqn.(7):

$$S(w_i, w_j) = \begin{cases} 1, & (\alpha_i \cdot \alpha_j^T)/(\|\alpha_i\| \cdot \|\alpha_j\|) \geq \delta \\ 0, & (\alpha_i \cdot \alpha_j^T)/(\|\alpha_i\| \cdot \|\alpha_j\|) < \delta \end{cases} \quad (7)$$

If $S(w_i, w_j) = 1$, w_i and w_j should be segmented together, otherwise, w_i and w_j belong to different semantic concepts respectively. Here, we denote the total number of segments of the query as integer m.

As mentioned in Section 2.4, m should be equal to k, therefore, the threshold δ is modified by k and m. We set the initial value $\delta = 0.5$ and modify it with binary search method until $m = k$. If k is larger than m, it means δ is too small to be a proper threshold, i.e. some segments should be further segmented. Otherwise, δ is too large that it should be reduced.

3 Experiments

3.1 Data set

We experiment on the data set published by (Bergsma and Wang, 2007). This data set comprises 500 queries which were randomly taken from the AOL search query database and each query. These queries are all segmented manually by three annotators (the results are referred as **A**, **B** and **C**).

We evaluate our results on the five test data sets (Tan and Peng, 2008), i.e. we use A, B, C, the intersection of three annotator's results (referred to as **D**) and the conjunction of three annotator's results (referred to as **E**). Besides, three evaluation metrics are used in our experiments (Tan and Peng, 2008; Peng and Schuurmans, 2001), i.e. *Precision* (referred to as Prec), *Recall and F-Measure* (referred to as F-mea).

3.2 Experimental results

Two baselines are used in our experiments: one is MI based method (referred to as MI), and the other is EM optimization (referred to as EM). Since the EM proposed in (Tan and Peng, 2008) is implemented with Yahoo! web corpus and only Google Soap Search API is available in our study, we adopt $t\text{-}test$ to evaluate the performance of MI with Google data (referred to as MI(G)) and Yahoo! web corpus (referred to as MI(Y)). With the values of MI(Y) and MI(G) in Table 1 we get the $p\text{-}value$ ($p = 0.316 \gg 0.05$), which indicates that the performance of MI with different corpuses has no significant difference. Therefore, we can deduce that, the two corpuses have little influence on the performance of the approaches. Here, we denote our approach as "ES", i.e. Eigenspace Similarity approach.

Table 1 presents the performance of the three approaches, i.e. MI (MI(Y) and MI(G)), EM and our proposed ES on the five test data sets using the three mentioned metrics. From Table 1 we find that ES achieves significant improvements as compared to the other two methods in any metric and data set we used.

For further analysis, we compute statistical performance on mathematical expectation and standard deviation as shown in Figure 2. We observe a consistent trend of the three metrics increasing from left to right as shown in Figure 2, i.e. EM performs better than MI and ES is the best among the three approaches.

		MI(Y)	MI(G)	EM	ES
A	Prec	0.469	0.548	0.562	0.652
	Recall	0.534	0.489	0.555	0.699
	F-mea	0.499	0.517	0.558	**0.675**
B	Prec	0.408	0.449	0.568	0.632
	Recall	0.472	0.391	0.578	0.659
	F-mea	0.438	0.418	0.573	**0.645**
C	Prec	0.451	0.503	0.558	0.614
	Recall	0.519	0.440	0.561	0.649
	F-mea	0.483	0.469	0.559	**0.631**
D	Prec	0.510	0.574	0.640	0.772
	Recall	0.550	0.510	0.650	0.826
	F-mea	0.530	0.540	0.645	**0.798**
E	Prec	0.582	0.672	0.715	0.834
	Recall	0.654	0.734	0.721	0.852
	F-mea	0.616	0.702	0.718	**0.843**

Table 1: Performance of different approaches.

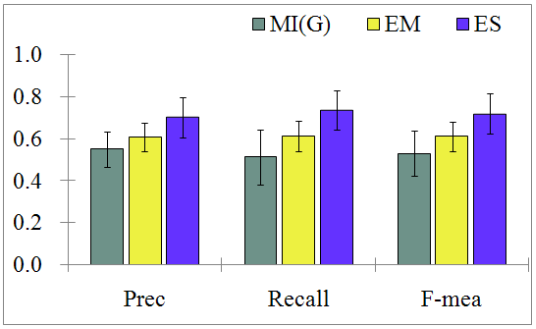


Figure 2: Statistical performance of approaches

First, we observe that, EM (Prec: 0.609, Recall: 0.613, F-mea: 0.611) performs much better than MI (Prec: 0.549, Recall: 0.513, F-mea: 0.529). This is because EM optimizes the frequencies of query words with EM algorithms. In addition, it should be noted that, the recall of MI is especially unsatisfactory, which is caused by its shortcoming on handling long entities.

Second, when compared with EM, ES also has more than 15% increase in the three reference metrics (15.1% on Prec, 20.2% on Recall and 17.7% on F-mea). Here all increases are statistically significant with $p\text{-}value$ closed to 0. In depth analysis indicates that this is because ES makes good use of the frequencies of query words in its principal eigenspace, while EM algorithm trains the observed data (frequencies of query words) by simply maximizing them using maximum likelihood.

4 Conclusion and Future work

We proposed an unsupervised approach for query segmentation. After using n-gram model to estimate term frequency matrix using term occurrence statistics from the web, we explored a new method to select principal eigenvectors and calculate the similarities of query words for segmentation. Experiments demonstrated the effectiveness of our approach, with significant improvement in segmentation accuracy as compared to the previous works.

Our approach will be capable of extracting semantic concepts from queries. Besides, it can extended to Chinese word segmentation. In future, we will further explore a new method of parameter k selection to achieve higher performance.

References

S. Bergsma and Q. I. Wang. 2007. *Learning Noun Phrase Query Segmentation*. In Proc of EMNLP-CoNLL

R. Jones, B. Rey, O. Madani, and W. Greiner. 2006. *Generating query substitutions*. In Proc of WWW.

I.T. Jolliffe. 2002. *Principal Component Analysis*. Springer, NY, USA.

Andrew Y. Ng, Michael I. Jordan, Yair Weiss. 2001. *On spectral clustering: Analysis and an algorithm* In Proc of NIPS.

F. Peng and D. Schuurmans. 2001. *Self-Supervised Chinese Word Segmentation*. Proc of the 4th Int'l Conf. on Advances in Intelligent Data Analysis.

K. M. Risvik, T. Mikolajewski, and P. Boros. 2003. *Query Segmentation for Web Search*. In Proc of WWW.

Bin Tan, Fuchun Peng. 2008. *Unsupervised Query Segmentation Using Generative Language Models and Wikipedia.* In Proc of WWW.

W. J. Teahan Rodger Mcnab Yingying Wen Ian H. Witten . 2000. *A compression-based algorithm for Chinese word segmentation* Computational Linguistics.

Xin-Jing Wang, Wen Liu, Yong Qin. 2007. *A Search-based Chinese Word Segmentation Method.* In Proc of WWW.

Yair Weiss. 1999. *Segmentation using eigenvectors: a unifying view.* Proc. IEEE Int'l Conf. Computer Vision, vol. 2, pp. 975-982.

Jia Xu, Jianfeng Gao, Kristina Toutanova, Hermann. 2008. *Bayesian Semi-Supervised Chinese Word Segmentation for Statistical Machine Translation.* In Proc of COLING.

Learning Semantic Categories from Clickthrough Logs

Mamoru Komachi
Nara Institute of Science and Technology (NAIST)
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
mamoru-k@is.naist.jp

Shimpei Makimoto and **Kei Uchiumi** and **Manabu Sassano**
Yahoo Japan Corporation
Midtown Tower, 9-7-1 Akasaka, Minato-ku, Tokyo 107-6211, Japan
{smakimot,kuchiumi,msassano}@yahoo-corp.jp

Abstract

As the web grows larger, knowledge acquisition from the web has gained increasing attention. In this paper, we propose using web search clickthrough logs to learn semantic categories. Experimental results show that the proposed method greatly outperforms previous work using only web search query logs.

1 Introduction

Compared to other text resources, search queries more directly reflect search users' interests (Silverstein et al., 1998). Web search logs are getting a lot more attention lately as a source of information for applications such as targeted advertisement and query suggestion.

However, it may not be appropriate to use queries themselves because query strings are often too heterogeneous or inspecific to characterize the interests of the user population. Although it is not clear that query logs are the best source of learning semantic categories, all the previous studies using web search logs rely on web search query logs.

Therefore, we propose to use web search clickthrough logs to learn semantic categories. Joachims (2002) developed a method that utilizes clickthrough logs for training ranking of search engines. A *search clickthrough* is a link which search users click when they see the result of their search. The intentions of two distinct search queries are likely to be similar, if not identical, when they have the same clickthrough. Search clickthrough logs are thus potentially useful for learnin semantic categories. Clickthrough logs have the additional advantage that they are available in abundance and can be stored at very low cost.[1] Our proposed method employs search clickthrough logs to improve semantic category acquisition in both precision and recall.

We cast semantic category acquisition from search logs as the task of learning labeled instances from few labeled seeds. To our knowledge this is the first study that exploits search clickthrough logs for semantic category learning.[2]

2 Related Work

There are many techniques that have been developed to help elicit knowledge from query logs. These algorithms use contextual patterns to extract a category or a relation in order to learn a target *instance* which belongs to the category (e.g. *cat* in *animal* class) or a pair of words in specific relation (e.g. *headquarter* to a *company*). In this work, we focus on extracting named entities of the same class to learn semantic categories.

Paşca and Durme (2007) were the first to discover the importance of search query logs in natural language processing applications. They focused on learning attributes of named entities, and thus their objective is different from ours. Another line of new research is to combine various resources such as web documents with search query logs (Paşca and Durme, 2008; Talukdar et al., 2008). We differ from this work in that we use search clickthrough logs rather than search query logs.

Komachi and Suzuki (2008) proposed a bootstrapping algorithm called *Tchai*, dedicated to the task of semantic category acquisition from search query logs. It achieves state-of-the-art performance for this task, but it only uses web search query logs.

[1] As for data availability, MSN Search query logs (RFP 2006 dataset) were provided to WSCD09: Workshop on Web Search Click Data 2009 participants. http://research.microsoft.com/en-US/um/people/nickcr/WSCD09/

[2] After the submission of this paper, we found that (Xu et al., 2009) also applies search clickthrough logs to this task. This work independently confirms the effectiveness of clickthrough logs to this task using different sources.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 189–192,
Suntec, Singapore, 4 August 2009.

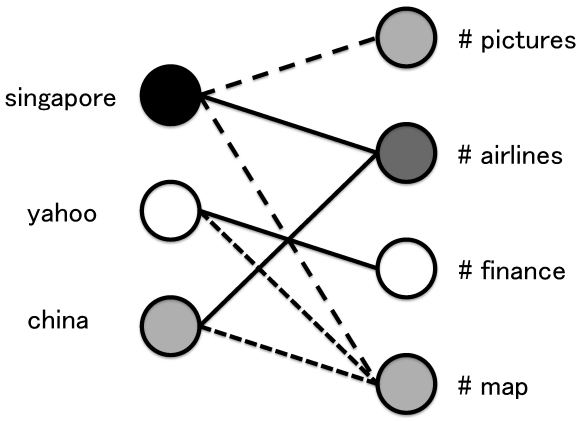


Figure 1: Labels of seeds are propagated to unlabeled nodes.

3 *Quetchup*[3] Algorithm

In this section, we describe an algorithm for learning semantic categories from search logs using label propagation. We name the algorithm *Quetchup*.

3.1 Semi-supervised Learning by Laplacian Label Propagation

Graph-based semi-supervised methods such as label propagation are known to achieve high performance with only a few seeds and have the advantage of scalability.

Figure 1 illustrates the process of label propagation using a seed term "singapore" to learn the Travel domain.

This is a bipartite graph whose left-hand side nodes are terms and right-hand side nodes are patterns. The strength of lines indicates relatedness between each node. The darker a node, the more likely it belongs to the Travel domain. Starting from "singapore," the pattern "♯ airlines" [4] is strongly related to "singapore," and thus the label of "singapore" will be propagated to the pattern. On the other hand, the pattern "♯ map" is a neutral pattern which co-occurs with terms other than the Travel domain such as "google" and "yahoo." Since the term "china" shares two patterns, "♯ airlines" and "♯ map," with "singapore," the label of the seed term "singapore" propagates to "china." "China" will then be classified in the Travel domain. In this way, label propagation gradually propagates the label of seed instances to neighbouring nodes, and optimal labels are given as the labels at which the label propagation process has converged.

Input:
Seed instance vector $F(0)$
Instance similarity matrix A
Output:
Instance score vector $F(t)$
1: Construct the normalized Laplacian matrix $L = I - D^{-1/2}AD^{-1/2}$
2: Iterate $F(t+1) = \alpha(-L)F(t) + (1-\alpha)F(0)$ until convergence

Figure 2: Laplacian label propagation algorithm

Figure 2 describes label propagation based on the *regularized Laplacian*. Let a sample x_i be $x_i \in \mathcal{X}$, $F(0)$ be a score vector of x comprised of a label set $y_i \in \mathcal{Y}$, and $F(t)$ be a score vector of x after step t. *Instance-instance similarity matrix* A is defined as $A = W^T W$ where $W$is a row-normalized *instance-pattern matrix*. The (i, j)-th element of W_{ij} contains the normalized frequency of co-occurrence of instance x_i and pattern p_j. D is a diagonal degree matrix of N where the (i, i)th element of D is given as $D_{ii} = \sum_j N_{ij}$.

This algorithm in Figure 2 is similar to (Zhou et al., 2004) except for the method of constructing A and the use of graph Laplacian. Zhou et al. proposed a heuristic to set $A_{ii} = 0$ to avoid self-reinforcement[5] because Gaussian kernel was used to create A. The Laplacian label propagation does not need such a heuristic because the graph Laplacian automatically reduces self-reinforcement by assigning negative weights to self-loops.

In the task of learning one category, scores of labeled (seed) instances are set to 1 whereas scores of unlabeled instances are set to 0. The output is a score vector which holds relatedness to seed instances in descending order. In the task of learning two categories, scores of seed instances are set to either 1 or -1, respectively, and the final label of instance x_i will be determined by the sign of output score vector y_i.

Label propagation has a parameter $\alpha \in (0, 1]$ that controls how much the labels of seeds are emphasized. As α approaches 0 it puts more weight on labeled instances, while as α increases it employs both labeled and unlabeled data.

There exists a closed-form solution for Laplacian label propagation:

[3] Query Term Chunk Processor

[4] ♯ is the place into which a query fits.

[5] Avoiding self-reinforcement is important because it causes semantic drift, a phenomenon where frequent instances and patterns unrelated to seed instances infect semantic category acquisition as iteration proceeds.

Category	Seed
Travel	jal (Japan Airlines), ana (All Nippon Airways), jr (Japan Railways), じゃらん (jalan: online travel guide site), his (H.I.S.Co.,Ltd.: travel agency)
Finance	みずほ銀行 (Mizuho Bank), 三井住友銀行 (Sumitomo Mitsui Banking Corporation), jcb, 新生銀行 (Shinsei Bank), 野村證券 (Nomura Securities)

Table 1: Seed terms for each category

$$F^* = \sum_{t=0}^{\infty}(\alpha(-L))^t F(0) = (I + \alpha L)^{-1} F(0)$$

However, the matrix inversion leads to $O(n^3)$ complexity, which is far from realistic in a real-world configuration. Nonetheless, it can be approximated by fixing the number of steps for label propagation.

4 Experiments with Web Search Logs

We will describe experimental result comparing a previous method *Tchai* to the proposed method *Quetchup* with clickthrough logs (*Quetchup*$_{click}$) and with query logs (*Quetchup*$_{query}$).

4.1 Experimental Settings

Search logs We used Japanese search logs collected in August 2008 from Yahoo! JAPAN Web Search. We thresholded both search query and clickthrough logs and retained the top 1 million distinct queries. Search logs are accompanied by their frequencies within the logs.

Construction of an instance-pattern matrix We used clicked links as clickthrough patterns. Links clicked less than 200 times were removed. After that, links which had only one co-occurring query were pruned. [6] On the other hand, we used two term queries as contextual patterns. For instance, if one has the term "singapore" and the query "singapore airlines," the contextual pattern "♯ airlines" will be created. Query patterns appearing less than 100 times were discarded.

The (i, j)-th element of a row-normalized instance-pattern matrix W is given by $W_{ij} = \frac{|x_i, p_j|}{\sum_k |x_i, p_k|}$.

Target categories We used two categories, Travel and Finance, to compare proposed methods with (Komachi and Suzuki, 2008).

When a query was a variant of a term or contains spelling mistakes, we estimated original form and manually assigned a semantic category. We allowed a query to have more than two categories. When a query had more than two terms, we assigned a semantic category to the whole query taking each term into account.[7]

System We used the same seeds presented in Table 1 for both *Tchai* and *Quetchup*. We used the same parameter for *Tchai* described in (Komachi and Suzuki, 2008), and collected 100 instances by iterating 10 times and extracting 10 instances per iteration. The number of iteration of *Quetchup* is set to 10. The parameter α is set to 0.0001.

Evaluation It is difficult in general to define recall for the task of semantic category acquisition since the true set of instances is not known. Thus, we evaluated all systems using *precision at* k and *relative recall* (Pantel and Ravichandran, 2004).[8] Relative recall is the coverage of a system given another system as baseline.

4.2 Experimental Result

4.2.1 Effectiveness of Clickthrough Logs

Figures 3 to 6 plot precision and relative recall for three systems to show effectiveness of search clickthrough logs in improvement of precision and relative recall. Relative recall of *Quetchup*$_{click}$ and *Tchai* were calculated against *Quetchup*$_{query}$.

Quetchup$_{click}$ gave the best precision among three systems, and did not degenerate going down through the list. In addition, it was demonstrated that *Quetchup*$_{click}$ gives high recall. This result shows that search clickthrough logs effectively improve both precision and recall for the task of semantic category acquisition.

On the other hand, *Quetchup*$_{query}$ degraded in precision as its rank increased. Manual check of the extracted queries revealed that the most prominent queries were Pornographic queries, followed by Food, Job and Housing, which frequently appear in web search logs. Other co-occurrence metrics such as pointwise mutual information would be explored in the future to suppress the effect of frequent queries.

In addition, *Quetchup*$_{click}$ constantly outperformed *Tchai* in both the Travel and Fi-

[6] Pruning facilitates the computation time and reduces the size of instance-pattern matrix drastically.

[7] Since web search query logs contain many spelling mistakes, we experimented in a realistic configuration.

[8] Typically, precision at k is the most important measure since the top k highest scored terms are evaluated by hand.

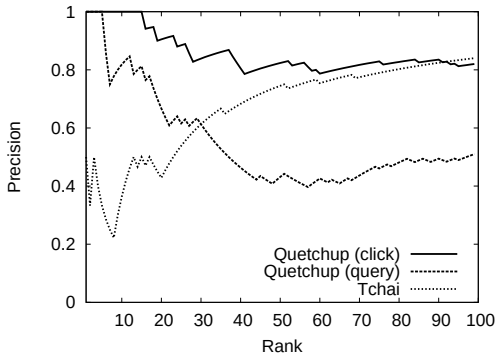


Figure 3: Precision of Travel domain

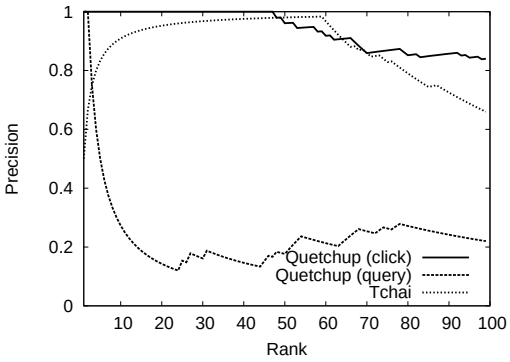


Figure 4: Precision of Finance domain

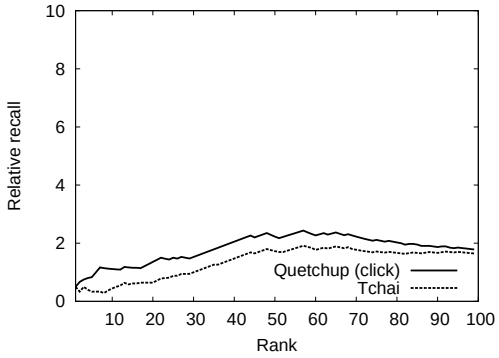


Figure 5: Relative recall of Travel domain

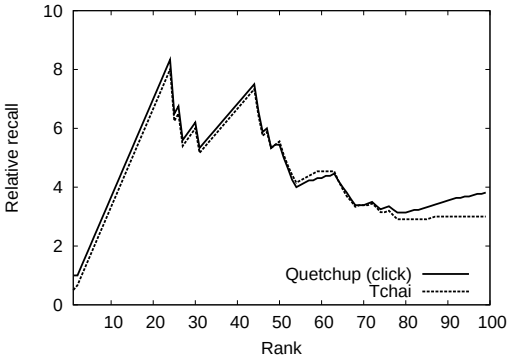


Figure 6: Relative recall of Finance domain

nance domains in precision and outperfomed *Quetchup*$_{query}$ in relative recall. The differences between the two domains of query-based systems seem to lie in the size of correct instances. The Finance domain is a closed set which has only a few effective query patterns, whereas Travel domain is an open set which has many query patterns that match correct instances. *Quetchup*$_{click}$ has an additional advantage that it is stable across over the ranked list, because the variance of the number of clicked links is small thanks to the nature of the ranking algorithm of search engines.

5 Conclusion

We have proposed a method called *Quetchup* to learn semantic categories from search clickthrough logs using Laplacian label propagation. The proposed method greatly outperforms previous method, taking the advantage of search clickthrough logs.

Acknowledgements

The first author is partly supported by the grant-in-aid JSPS Fellowship for Young Researchers. We thank the anonymous reviewers for helpful comments and suggestions.

References

T. Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. *KDD*, pages 133–142.

M. Komachi and H. Suzuki. 2008. Minimally Supervised Learning of Semantic Knowledge from Query Logs. *IJCNLP*, pages 358–365.

M. Paşca and B. V. Durme. 2007. What You Seek is What You Get: Extraction of Class Attributes from Query Logs. *IJCAI-07*, pages 2832–2837.

M. Paşca and B. V. Durme. 2008. Weakly-Supervised Acquisition of Open-Domain Classes and Class Attributes from Web Documents and Query Logs. *ACL-2008*, pages 19–27.

P. Pantel and D. Ravichandran. 2004. Automatically Labeling Semantic Classes. *HLT/NAACL-04*, pages 321–328.

C. Silverstein, M. Henzinger, H. Marais, and M. Moricz. 1998. *Analysis of a Very Large AltaVista Query Log*. Digital SRC Technical Note 1998-014.

P. P. Talukdar, J. Reisinger, M. Paşca, D. Ravichandran, R. Bhagat, and F. Pereira. 2008. Weakly-Supervised Acquisition of Labeled Class Instances using Graph Random Walks. *EMNLP-2008*, pages 581–589.

G. Xu, S. Yang, and H. Li. 2009. Named Entity Mining from Click-Through Log Using Weakly Supervised Latent Dirichlet Allocation. *KDD*. to appear.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schökopf. 2004. Learning with Local and Global Consistency. *NIPS*, 16:321–328.

A Rose is a Roos is a Ruusu: Querying Translations for Web Image Search

Janara Christensen **Mausam** **Oren Etzioni**
Turing Center
Dept. of Computer Science and Engineering
University of Washington, Seattle, WA 98105 USA
{janara, mausam, etzioni} @cs.washington.edu

Abstract

We query Web Image search engines with words (*e.g.*, spring) but need images that correspond to particular senses of the word (*e.g.*, flexible coil). Querying with polysemous words often yields unsatisfactory results from engines such as Google Images. We build an image search engine, IDIOM, which improves the quality of returned images by focusing search on the desired *sense*. Our algorithm, instead of searching for the original query, searches for multiple, automatically chosen translations of the sense in several languages. Experimental results show that IDIOM outperforms Google Images and other competing algorithms returning 22% more relevant images.

1 Introduction

One out of five Web searches is an image search (Basu, 2009). A large subset of these searches is subjective in nature, where the user is looking for different images for a single concept (Linsley, 2009). However, it is a common user experience that the images returned are not relevant to the intended concept. Typical reasons include (1) existence of homographs (other words that share the same spelling, possibly in another language), and (2) polysemy, several meanings of the query word, which get merged in the results.

For example, the English word 'spring' has several senses – (1) the season, (2) the water body, (3) spring coil, and (4) to jump. Ten out of the first fifteen Google images for *spring* relate to the season sense, three to water body, one to coil and none to the jumping sense. Simple modifications to query do not always work. Searching for *spring water* results in many images of bottles of spring water and searching for *spring jump* returns only three images (out of fifteen) of someone jumping.

Polysemous words are common in English. It is estimated that average polysemy of English is more than 2 and average polysemy of common English words is much higher (around 4). Thus, it is not surprising that polysemy presents a significant limitation in the context of Web Search. This is especially pronounced for image search where query modification by adding related words may not help, since, even though the new words might be present on the page, they may not be all associated with an image.

Recently Etzioni *et al.* (2007) introduced PANIMAGES, a novel approach to image search, which presents the user with a set of translations. *E.g.*, it returns 38 translations for the coil sense of spring. The user can query one or more translations to get the relevant images. However, this method puts the onus of choosing a translation on the user. A typical user is unaware of most properties of languages and has no idea whether a translation will make a good query. This results in an added burden on the user to try different translations before finding the one that returns the relevant images.

Our novel system, IDIOM, removes this additional burden. Given a desired sense it *automatically* picks the good translations, searches for associated images and presents the final images to the user. For example, it automatically queries the French *ressort* when looking for images of spring coil. We make the following contributions:

- We automatically learn a predictor for "good" translations to query given a desired sense. A good translation is one that is monosemous and is in a major language, *i.e.*, is expected to yield a large number of images.
- Given a sense we run our predictor on all its translations to shortlist a set of three translations to query.
- We evaluate our predictor by comparing the images that its shortlists return against the

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 193–196,
Suntec, Singapore, 4 August 2009.

images that several competing methods return. Our evaluation demonstrates that IDIOM returns at least one good image for 35% more senses (than closest competitor) and overall returns 22% better images.

2 Background

IDIOM makes heavy use of a sense disambiguated, vastly multilingual dictionary called PANDICTIONARY (Mausam et al., 2009). PANDICTIONARY is automatically constructed by probabilistic inference over a graph of translations, which is compiled from a large number of multilingual and bilingual dictionaries. For each sense PANDICTIONARY provides us with a set of translations in several languages. Since it is generated by inference, some of the asserted translations may be incorrect – it additionally associates a probability score with each translation. For our work we choose a probability threshold such that the overall precision of the dictionary is 0.9 (evaluated based on a random sample). PANDICTIONARY has about 80,000 senses and about 1.8 million translations at precision 0.9.

We use Google Image Search as our underlying image search engine, but our methods are independent of the underlying search engine used.

3 The IDIOM Algorithm

At the highest level IDIOM operates in three main steps: (1) Given a new query q it looks up its various senses in PANDICTIONARY. It displays these senses and asks the user to select the intended sense, s_q. (2) It runs Algorithm 1 to shortlist three translations of s_q that are expected to return high quality images. (3) It queries Google Images using the three shortlisted translations and displays the images. In this fashion IDIOM searches for images that are relevant to the intended concept as opposed to using a possibly ambiguous query.

The key technical component is the second step – shortlisting the translations. We first use PANDICTIONARY to acquire a set of high probability translations of s_q. We run each of these translations through a learned classifier, which predicts whether it will make a good query, *i.e.*, whether we can expect images relevant to this sense if queried using this translation. The classifier additionally outputs a confidence score, which we use to rank the various translations. We pick the top three translations, as long as they are above a minimum confidence score, and return those as the shortlisted queries. Algorithm 1 describes this as a pseudo-code.

Algorithm 1 findGoodTranslationsToQuery(s_q)

```
translations = translations of s_q in PANDICTIONARY
for all w ∈ translations do
    pd = getPanDictionaryFeatures(w, s_q)
    g = getGoogleFeatures(w, s_q)
    conf[w] = confidence in Learner.classify(pd, g)
sort all words w in decreasing order of conf scores
return top three w from the sorted list
```

3.1 Features for Classifier

What makes a translation w good to query? A desired translation is one that (1) is in a high-coverage language, so that the number of images returned is large, (2) monosemously expresses the intended sense s_q, or at least has this sense as its dominant sense, and (3) does not have homographs in other languages. Such a translation is expected to yield images relevant to only the intended sense. We construct several features that provide us evidence for these desired characteristics. Our features are automatically extracted from PANDICTIONARY and Google.

For the first criterion we restrict the translations to a set of high-coverage languages including English, French, German, Spanish, Chinese, Japanese, Arabic, Russian, Korean, Italian, and Portuguese. Additionally, we include the *language* as well as *number of documents returned by Google search of* w as features for the classifier.

To detect if w is monosemous we add a feature reflecting the degree of polysemy of w: the *number of* PANDICTIONARY *senses that* w *belongs to*. The higher this number the more polysemous w is expected to be. We also include the *number of languages that have* w *in their vocabulary*, thus, adding a feature for the degree of homography.

PANDICTIONARY is arranged such that each sense has an English source word. If the source word is part of many senses but s_q is much more popular than others or s_q is ordered before the other senses then we can expect s_q to be the dominant sense for this word. We include features like *size of the sense* and *order of the sense*.

Part of speech of s_q is another feature. Finally we also add the *probability score* that w is a translation of s_q in our feature set.

3.2 Training the Classifier

To train our classifier we used Weka (Witten and Frank, 2005) on a hand labeled dataset of 767 ran-

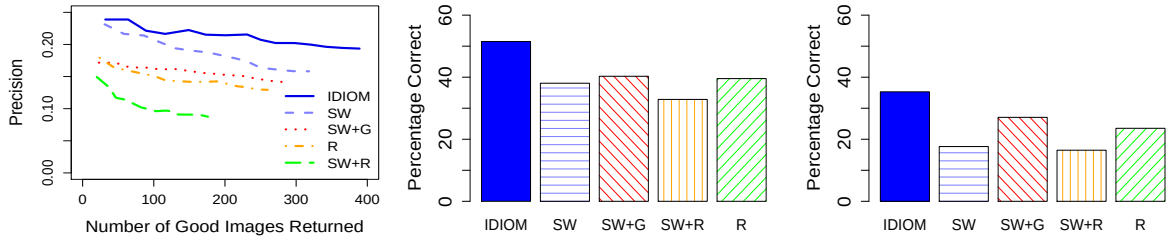


Figure 1: (a): Precision of images *vs.* the number of relevant images returned. IDIOM covers the maximum area. (b,c) The percentage of senses for which at least one relevant result was returned, for (b) all senses and (c) for minor senses of the queries.

domly chosen word sense pairs (*e.g.*, pair of 'primavera,' and 'the season spring'). We labeled a pair as positive if googling the word returns at least one good image for the sense in the top three. We compared performance among a number of machine learning algorithms and found that Random Forests (Breiman, 2001) performed the best overall with 69% classification accuracy using ten fold cross validation versus 63% for Naive Bayes and 62% for SVMs. This high performance of Random Forests mirrors other past experiments (Caruana and Niculescu-Mizil, 2006).

Because of the ensemble nature of Random Forests it is difficult to inspect the learned classifier for analysis. Still, anecdotal evidence suggests that the classifier is able to learn an effective model of good translations. We observe that it favors English whenever the English word is part of one or few senses – it picks out *auction* when the query is 'sale' in the sense of "act of putting up for auction to highest bidder". In cases where English is more ambiguous it chooses a relatively less ambiguous word in another language. It chooses the French word *ressort* for finding 'spring' in the sense of coil. For the query 'gift' we notice that it does not choose the original query. This matches our intuition, since gift has many homographs – the German word 'Gift' means poison or venom.

4 Experiments

Can querying translations instead of the original query improve the quality of image search? If so, then how much does our classifier help compared to querying random translations? We also analyze our results and study the variation of image quality along various dimensions, like part of speech, abstractness/concreteness of the sense, and ambiguity of the original query.

As a comparison, we are interested in how IDIOM performs in relation to other methods for querying Google Images. We compare IDIOM to several methods. (1) *Source Word* (SW): Querying with only the source word. This comparison functions as our baseline. (2) *Source Word + Gloss* (SW+G): Querying with the source word and the gloss for the sense[1]. This method is one way to focus the source word towards the desired sense. (3) *Source Word + Random* (SW+R): Querying with three pairs of source word and a random translation. This is another natural way to extend the baseline for the intended sense. (4) *Random* (R): Querying with three random translations. This tests the extent to which our classifier improves our results compared to randomly choosing translations shown to the user in PANIMAGES.

We randomly select fifty English queries from PANDICTIONARY and look up all senses containing these in PANDICTIONARY, resulting in a total of 134 senses. These queries include short word sequences (*e.g.*, 'open sea'), mildly polysemous queries like 'pan' (means Greek God and cooking vessel) and highly polysemous ones like 'light'.

For each sense of each word, we query Google Images with the query terms suggested by each method and evaluate the top fifteen results. For methods in which we have three queries, we evaluate the top five results for each query. We evaluate a total of fifteen results because Google Images fits fifteen images on each page for our screen size.

Figure 1(a) compares the precision of the five methods with the number of good images returned. We vary the number of images in consideration from 1 to 15 to generate various points in the graph. IDIOM outperforms the others by wide margins overall producing a larger number of good images and at higher precision. Surprisingly, the closest competitor is the baseline method as opposed to other methods that try to focus the search towards the intended sense. This is probably because the additional words in the query (either from gloss or a random translation) confuse Google Images rather than focusing the search. IDIOM covers 41% more area than SW. Overall

[1]PANDICTIONARY provides a gloss (short explanation) for each sense. *E.g.*, a gloss for 'hero' is 'role model.'

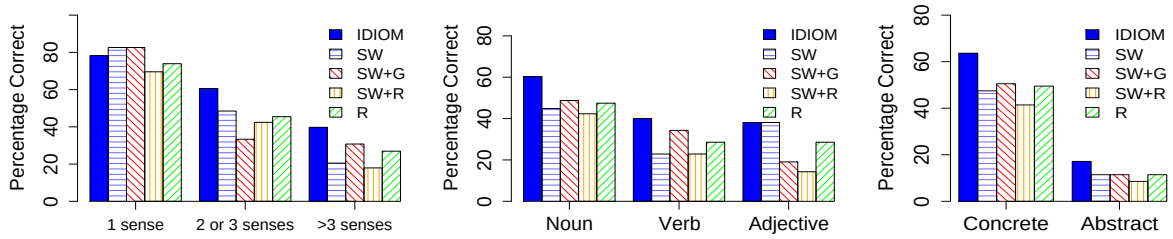


Figure 2: The percentage of senses for which at least one relevant result was returned varied along several dimensions: (a) polysemy of original query, and (b) part of speech of the sense, (c) abstractness/concreteness of the sense.

IDIOM produces 22% better images compared to SW (389 vs 318).

We also observe that random translations return much worse images than IDIOM suggesting that a classifier is essential for high quality images.

Figure 1(b) compares the percentage of senses for which at least one good result was returned in the fifteen. Here IDIOM performs the best at 51%. Each other method performs at about 40%. The results are statistically highly significant ($p < 0.01$).

Figure 1(c) compares the performance just on the subset of the non-dominant senses of the query words. All methods perform worse than in Figure 1(b) but IDIOM outperforms the others.

We also analyze our results across several dimensions. Figure 2(a) compares the performance as a function of polysemy of the original query. As expected, the disparity in methods is much more for high polysemy queries. Most methods perform well for the easy case of unambiguous queries.

Figure 2(b) compares along the different parts of speech. For nouns and verbs, IDIOM returns the best results. For adjectives, IDIOM and SW perform the best. Overall, nouns are the easiest for finding images and we did not find much difference between verbs and adjectives.

Finally, Figure 2(c) reports how the methods perform on abstract versus concrete queries. We define a sense as abstract if it does not have a natural physical manifestation. For example, we classify 'nest' (a bird built structure) as concrete, and 'confirm' (to strengthen) as abstract. IDIOM performs better than the other methods, but the results vary massively between the two categories.

Overall, we find that our new system consistently produces better results across the several dimensions and various metrics.

5 Related Work and Conclusions

Related Work: The popular paradigm for image search is keyword-based, but it suffers due to polysemy and homography. An alternative paradigm is content based (Datta et al., 2008), which is very slow and works on simpler images. The field of cross-lingual information retrieval (Ballesteros and Croft, 1996) often performs translation-based search. Other than PANIMAGES (which we outperform), no one to our knowledge has used this for image search.

Conclusions: The recent development of PANDICTIONARY (Mausam et al., 2009), a sense-distinguished, massively multilingual dictionary, enables a novel image search engine called IDIOM. We show that querying unambiguous translations of a sense produces images for 35% more concepts compared to querying just the English source word. In the process we learn a classifier that predicts whether a given translation is a good query for the intended sense or not. We plan to release an image search website based on IDIOM. In the future we wish to incorporate knowledge from WordNet and cross-lingual links in Wikipedia to increase IDIOM's coverage beyond the senses from PANDICTIONARY.

References

L. Ballesteros and B. Croft. 1996. Dictionary methods for cross-lingual information retrieval. In *DEXA Conference on Database and Expert Systems Applications*.

Dev Basu. 2009. How To Leverage Rich Media SEO for Small Businesses. In *Search Engine Journal. http://www.searchenginejournal.com/rich - media-small-business-seo/9580.*

L. Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

R. Caruana and A. Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In *ICML'06*, pages 161–168.

R. Datta, D. Joshi, J. Li, and J. Wang. 2008. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60.

O. Etzioni, K. Reiter, S. Soderland, and M. Sammer. 2007. Lexical translation with application to image search on the Web. In *Machine Translation Summit XI.*

Peter Linsley. 2009. Google Image Search. In *SMX West.*

Mausam, S. Soderland, O. Etzioni, D. Weld, M. Skinner, and J. Bilmes. 2009. Compiling a massive, multilingual dictionary via probabilistic inference. In *ACL'09*.

I. Witten and E. Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

Extracting Paraphrases of Technical Terms from Noisy Parallel Software Corpora

Xiaoyin Wang[1,2], **David Lo**[1], **Jing Jiang**[1], **Lu Zhang**[2], **Hong Mei**[2]
[1]School of Information Systems, Singapore Management University, Singapore, 178902
{xywang, davidlo, jingjiang}@smu.edu.sg
[2]Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education
Beijing, 100871, China
{zhanglu, meih}@sei.pku.edu.cn

Abstract

In this paper, we study the problem of extracting technical paraphrases from a parallel software corpus, namely, a collection of duplicate bug reports. Paraphrase acquisition is a fundamental task in the emerging area of text mining for software engineering. Existing paraphrase extraction methods are not entirely suitable here due to the noisy nature of bug reports. We propose a number of techniques to address the noisy data problem. The empirical evaluation shows that our method significantly improves an existing method by up to 58%.

1 Introduction

Using natural language processing (NLP) techniques to mine software corpora such as code comments and bug reports to assist software engineering (SE) is an emerging and promising research direction (Wang et al., 2008; Tan et al., 2007). Paraphrase extraction is one of the fundamental problems that have not been addressed in this area. It has many applications including software ontology construction and query expansion for retrieving relevant technical documents.

In this paper, we study automatic paraphrase extraction from a large collection of software bug reports. Most large software projects have bug tracking systems, e.g., Bugzilla[1], to help global users to describe and report the bugs they encounter when using the software. However, since the same bug may be seen by many users, many duplicate bug reports are sent to bug tracking systems. The duplicate bug reports are manually tagged and associated to the original bug report by either the system manager or software developers. These families of duplicate bug reports form a semi-parallel corpus and therefore a good candidate for extraction of paraphrases of technical terms. Hence, bug reports interest us because (1) they are abundant and freely available,(2) they naturally form a semi-parallel corpus, and (3) they contain many technical terms.

Parallel bug reports with a pair of true paraphrases	
1:	connector extend with a straight line in *full screen mode*
2:	connector show straight line in *presentation mode*
Non-parallel bug reports referring to the same bug	
1:	Settle language for part of text and spellchecking part of text
2:	Feature requested to improve the management of a multi-language document
Context-peculiar paraphrases (shown in italics)	
1:	status bar appear in the middle of *the screen*
2:	maximizing window create phantom status bar in middle of *document*

Table 1: Bug Report Examples

However, bug reports have characteristics that raise many new challenges. Different from many other parallel corpora, bug reports are *noisy*. We observe at least three types of noise common in bug reports. First, many bug reports have many spelling, grammatical and sentence structure errors. To address this we extend a suitable state-of-the-art technique that is robust to such corpora, i.e. (Barzilay and McKeown, 2001). Second, many duplicate bug report families contain sentences that are not truly parallel. An example is shown in Table 1 (middle). We handle this by considering lexical similarity between duplicate bug reports. Third, even if the bug reports are parallel, we find many cases of *context-peculiar paraphrases*, i.e., a pair of phrases that have the same meaning in a very narrow context. An example is shown in Table 1 (bottom). To address this, we introduce two notions of *global context-based score* and *co-occurrence based score* which take into account all good and bad occurrences of the phrases in a candidate paraphrase in the corpus. These scores are then used to identify and remove

[1]http://www.bugzilla.org/

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 197–200,
Suntec, Singapore, 4 August 2009.

context-peculiar paraphrases.

The contributions of our work are twofold. First, we studied the important problem of paraphrase extraction from a noisy semi-parallel software corpus, which has not been studied either in the NLP or the SE community. Second, taking into consideration the special characteristics of our noisy data, we proposed several improvements to an existing general paraphrase extraction method, resulting in a significant performance gain – up to 58% relative improvement in precision.

2 Related Work

In the area of text mining for software engineering, paraphrases have been used in many tasks, e.g., (Wang et al., 2008; Tan et al., 2007). However, most paraphrases used are obtained manually. A recent study using synonyms from WordNet highlights the fact that these are not effective in software engineering tasks due to domain specificity (Sridhara et al., 2008). Therefore, an automatic way to derive technical paraphrases specific to software engineering is desired.

Paraphrases can be extracted from non-parallel corpora using contextual similarity (Lin, 1998). They can also be obtained from parallel corpora if such data is available (Barzilay and McKeown, 2001; Ibrahim et al., 2003). Recently, there are also a number of studies that extract paraphrases from multilingual corpora (Bannard and Callison-Burch, 2005; Zhao et al., 2008).

The approach in (Barzilay and McKeown, 2001) does not use deep linguistic analysis and therefore is suitable to noisy corpora like ours. Due to this reason, we build our technique on top of theirs. The following provides a summary of their technique.

Two types of paraphrase patterns are defined: (1) Syntactic patterns which consist of the POS tags of the phrases. For example, the paraphrases "a VGA monitor" and "a monitor" are represented as "DT_1 JJ NN_2" $\leftrightarrow$ "DT_1 NN_2", where the subscripts denote common words. (2) Contextual patterns which consist of the POS tags before and after the phrases. For example, the contexts "in the middle of" and "in middle of" in Table 1 (bottom) are represented as "IN_1 DT NN_2 IN_3" $\leftrightarrow$ "IN_1 NN_2 IN_3".

During pre-processing, the parallel corpus is aligned to give a list of parallel sentence pairs. The sentences are then processed by a POS tagger and a chunker. The authors first used identical words and phrases as seeds to find and score contextual patterns. The patterns are scored based on the following formula: *(n+)/n*, in which, *n+* refers to the number of positively labeled paraphrases satisfying the patterns and *n* refers to the number of all paraphrases satisfying the patterns. Only patterns with scores above a threshold are considered. More paraphrases are identified using these contextual patterns, and more patterns are then found and scored using the newly-discovered paraphrases. This co-training algorithm is employed in an iterative fashion to find more patterns and positively labeled paraphrases.

3 Methodology

Our paraphrase extraction method consists of three components: sentence selection, global context-based scoring and co-occurrence-based scoring. We marry the three components together into a holistic solution.

Selection of Parallel Sentences Our corpus consists of short bug report summaries, each containing one or two sentences only, grouped by the bugs they report. Each group corresponds to reports pertaining to a single bug and are duplicate of one another. Therefore, reports belonging to the same group can be naturally regarded as parallel sentences.

However, these sentences are only partially parallel because two users may describe the same bug in very different ways. An example is shown in Table 1 (middle). This kind of sentence pairs should not be regarded as parallel. To address this problem, we take a heuristic approach and only select sentence pairs that have strong similarities. Our similarity score is based on the number of common words, bigrams and trigrams shared between two parallel sentences. We use a threshold of 5 to filter out non-parallel sentences.

Global Context-Based Scoring Our context-based paraphrase scoring method is an extension of (Barzilay and McKeown, 2001) described in Sec. 2. Parallel bug reports are usually noisy. At times, some words might be detected as paraphrases incidentally due to the noise. In (Barzilay and McKeown, 2001), a paraphrase is reported as long as there is a *single* good supporting pair of sentences. Although this works well for a relatively clean parallel corpus considered in their work, i.e., novels, this does not work well for bug reports. Consider the context-peculiar example in Table 1 (bottom). For a context-peculiar para-

phrase, there can be many sentences containing the pair of phrases but very few support them to be a paraphrase. We develop a technique to offset this noise by computing a *global* context-based score for two phrases being a paraphrase over *all* their parallel occurrences. This is defined by the following formula: $S_g = \frac{1}{n}\Sigma_{i=1}^{n} s_i$, where n is the number of parallel bug reports with the two phrases occurring in parallel, and s_i is the score for the i'th occurrence. s_i is computed as follows:

1. We compute the set of patterns with affixed pattern scores based on (Barzilay and McKeown, 2001).
2. For the i'th parallel occurrence of the pair of phrases we want to score, we try to find a pattern that matches the occurrence and assign the pattern score to the pair of phrases as s_i. If no such pattern exists, we set s_i to 0.

By taking the average of s_i as the global score for a pair of phrases, we do not rely much on a single s_i and can therefore prevent context-peculiar paraphrases to some degree.

Co-occurrence-Based Scoring We also consider another global co-occurrence-based score that is commonly used for finding collocations. A general observation is that noise tends to appear in random but random things do not occur in the same way often. It is less likely for randomly paired words or paraphrases to co-occur together many times. To compute the likelihood of two phrases occurring together, we use the following commonly used co-occurrence-based score:

$$S_c = \frac{P(w_1, w_2)}{P(w_1)P(w_2)}. \quad (1)$$

The expression $P(w_1, w_2)$ refers to the probability of a pair of phrases w_1 and w_2 appearing together. It is estimated based on the proportion of the corpus containing both w_1 and w_2 in parallel. Similarly, $P(w_1)$ and $P(w_2)$ each corresponds to the probability of w_1 and w_2 appearing respectively. We normalize the S_c score to the range of 0 to 1 by dividing it with the size of the corpus.

Holistic Solution We employ the parallel sentence selection as a pre-processing step, and merge co-occurrence-based scoring with global context-based scoring. For each parallel sentence pairs, a chunker is used to get chunks from each sentence. All possible pairings of chunks are then formed. This set of chunk pairs are later fed to the method in (Barzilay and McKeown, 2001) to produce a set of patterns with affixed scores. With this we compute our global-context based scores. The co-occurrence based scores are computed following the approach described above.

Two thresholds are used and candidate paraphrases whose scores are below the respective thresholds are removed. Alternatively, one of the score is used as a filter, while the other is used to rank the candidates. The next section describes our experimental results.

4 Evaluation

Data Set Our bug report corpus is built from OpenOffice[2]. OpenOffice is a well-known open source software which has similar functionalities as Microsoft Office. We use the bug reports that are submitted before Jan 1, 2008. Also, we only use the summary part of the bug reports.

We build our corpus in the following steps. We collect a total of 13,898 duplicate bug reports from OpenOffice. Each duplicate bug report is associated to a master report—there is one master report for each unique bug. From this information, we create duplicate bug report groups where each member of a group is a duplicate of all other members in the same group. Finally, we extract duplicate bug report pairs by pairing each two members of each group. We get in total 53,363 duplicate bug report pairs.

As the first step, we employ parallel sentence selection, described in Sec. 3, to remove non-parallel duplicate bug report pairs. After this step, we find 5,935 parallel duplicate bug report pairs.

Experimental Setup The baseline method we consider is the one in (Barzilay and McKeown, 2001) without sentence alignment – as the bug reports are usually of one sentence long. We call it *BL*. As described in Sec. 2, *BL* utilizes a threshold to control the number of patterns mined. These patterns are later used to select paraphrases. In the experiment, we find that running *BL* using their default threshold of 0.95 on the 5,935 parallel bug reports only gives us 18 paraphrases. This number is too small for practical purposes. Therefore, we reduce the threshold to get more paraphrases. For each threshold in the range of 0.45-0.95 (step size: 0.05), we extract paraphrases and compute the corresponding precision.

In our approach, we first form chunk pairs from the 5,935 pairs of parallel sentences and then use the baseline approach at a low threshold to ob-

[2] http://www.openoffice.org/

tain patterns. Using these patterns we compute the global context-based scores S_g. We also compute the co-occurrence scores S_c. We rank and extract top-k paraphrases based on these scores. We consider 4 different methods: We can use either S_g or S_c to rank the discovered paraphrases. We call them *Rk-*S_g and *Rk-*S_c. We also consider using one of the scores for ranking and the other for filtering bad candidate paraphrases. A threshold of 0.05 is used for filtering. We call these two methods *Rk-*S_c*+Ft-*S_g and *Rk-*S_g*+Ft-*S_c. With ranked lists from these 4 methods, we can compute precision@k for the top-k paraphrases.

Results The comparison among these methods is plotted in Figure 1. From the figure we can see that our holistic approach using global-context score to rank and co-occurrence score to filter (i.e., *Rk-*S_g*+Ft-*S_c) has higher precision than the baseline approach (i.e., *BL*) in all ks. In general, the other holistic configuration (i.e., *Rk-*S_c*+Ft-*S_g) also works well for most of the ks considered. Interestingly, the graph shows that using only one of the scores alone (i.e., *Rk-*S_g and *Rk-*S_c) does not result in a significantly higher precision than the baseline approach. A holistic approach by merging global-context score and co-occurrence score is needed to yield higher precision.

In Table 2, we show some examples of the paraphrases our algorithm extracted from the bug report corpus. As we can see, most of the paraphrases are very technical and only make sense in the software domain. It demonstrates the effectiveness of our method.

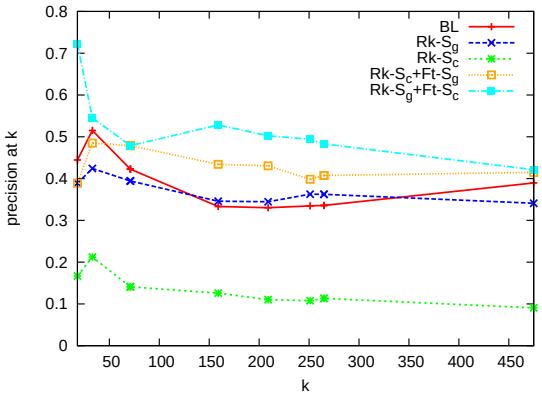


Figure 1: Precision@k for a range of k.

5 Conclusion

In this paper, we develop a new technique to extract paraphrases of technical terms from software bug reports. Paraphrases of technical terms have been shown to be useful for various software engineering tasks. These paraphrases could not be obtained via general purpose thesaurus e.g., WordNet. Interestingly, there is a wealth of text data, in particular bug reports, available for analysis in open-source software repositories. Despite their availability, a good technique is needed to extract paraphrases from these corpora as they are often noisy. We develop several approaches to address noisy data via parallel sentence selection, global-context based scoring and co-occurrence based scoring. To show the utility of our approach, we experimented with many parallel bug reports from a large software project. The preliminary experiment result is promising as it could significantly improves an existing method by up to 58%.

the edit-field	↔	input line field
presentation mode	↔	full screen mode
word separator	↔	a word delimiter
application	↔	app
freeze	↔	crash
mru file list	↔	recent file list
multiple monitor	↔	extended desktop
xl file	↔	excel file

Table 2: Examples of paraphrases of technical terms mined from bug reports.

References

C. Bannard and C. Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *ACL: Annual Meet. of Assoc. of Computational Linguistics*.

R. Barzilay and K. R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *ACL: Annual Meet. of Assoc. of Computational Linguistics*.

A. Ibrahim, B. Katz, and J. Lin. 2003. Extracting structural paraphrases from aligned monolingual corpora. In *Int. Workshop on Paraphrasing*.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *ACL: Annual Meet. of Assoc. of Computational Linguistics*.

G. Sridhara, E. Hill, L. Pollock, and K. Vijay-Shanker. 2008. Identifying word relations in software: A comparative study of semantic similarity tools. In *ICPC: Int. Conf. on Program Comprehension*.

L. Tan, D. Yuan, G. Krishna, and Y. Zhou. 2007. /*icomment: bugs or bad comments?*/. In *SOSP: Symp. on Operating System Principles*.

X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun. 2008. An approach to detecting duplicate bug reports using natural language and execution information. In *ICSE: Int. Conf. on Software Engineering*.

S. Zhao, H. Wang, T. Liu, and S. Li. 2008. Pivot approach for extracting paraphrase patterns from bilingual corpora. In *ACL: Annual Meet. of Assoc. of Computational Linguistics*.

Mining Association Language Patterns for Negative Life Event Classification

Liang-Chih Yu[1], Chien-Lung Chan[1], Chung-Hsien Wu[2] and Chao-Cheng Lin[3]
[1]Department of Information Management, Yuan Ze University, Taiwan, R.O.C.
[2]Department of CSIE, National Cheng Kung University, Taiwan, R.O.C.
[3]Department of Psychiatry, National Taiwan University Hospital, Taiwan, R.O.C.
{lcyu, clchan}@saturn.yzu.edu.tw, chwu@csie.ncku.edu.tw, linchri@gmail.com

Abstract

Negative life events, such as death of a family member, argument with a spouse and loss of a job, play an important role in triggering depressive episodes. Therefore, it is worth to develop psychiatric services that can automatically identify such events. In this paper, we propose the use of association language patterns, i.e., meaningful combinations of words (e.g., <loss, job>), as features to classify sentences with negative life events into predefined categories (e.g., Family, Love, Work). The language patterns are discovered using a data mining algorithm, called association pattern mining, by incrementally associating frequently co-occurred words in the sentences annotated with negative life events. The discovered patterns are then combined with single words to train classifiers. Experimental results show that association language patterns are significant features, thus yielding better performance than the baseline system using single words alone.

1 Introduction

With the increased incidence of depressive disorders, many psychiatric websites have developed community-based services such as message boards, web forums and blogs for public access. Through these services, individuals can describe their stressful or negative life events such as death of a family member, argument with a spouse and loss of a job, along with depressive symptoms, such as depressive mood, suicidal tendencies and anxiety. Such psychiatric texts (e.g., forum posts) contain large amounts of natural language expressions related to negative life events, making them useful resources for building more effective psychiatric services. For instance, a psychiatric retrieval service can retrieve relevant forum or blog posts according to the negative life events experienced by users so that they can be aware that they are not alone because many people have suffered from the same or similar problems. The users can then create a community discussion to share their experiences with each other. Additionally, a dialog system can generate supportive responses like "Don't worry", "That's really sad" and "Cheer up" if it can understand the negative life events embedded in the example sentences shown in Table 1. Therefore, this study proposes a framework for negative life event classification. We formulate this problem as a sentence classification task; that is, classify sentences according to the type of negative life events within them. The class labels used herein are presented in Table 1, which are derived from Brostedt and Pedersen (2003).

Traditional approaches to sentence classification (Khoo et al., 2006; Naughton et al., 2008) or text categorization (Sebastiani 2002) usually adopt bag-of-words as baseline features to train classifiers. Since the bag-of-words approach treats each word independently without considering the relationships of words in sentences, some researchers have investigated the use of *n*-grams to capture sequential relations between words to boost classification performance (Chitturi and Hansen, 2008; Li and Zong, 2008). The use of *n*-grams is effective in capturing local dependencies of words, but tends to suffer from data sparseness problem in capturing long-distance dependencies since higher-order *n*-grams require large training data to obtain reliable estimation. For our task, the expressions of negative life events can be characterized by *association language patterns*, i.e., meaningful combinations of words, such as <worry, children, health>, <break up, boyfriend>, <argue, friend>, <loss, job>, and

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 201–204,
Suntec, Singapore, 4 August 2009.

Label	Description	Example Sentence
Family	Serious illness of a family member; Son or daughter leaving home	I am very **worried** about my **children's** **health**.
Love	Spouse/mate engaged in infidelity; Broke up with a boyfriend or girlfriend	I **broke up** with my dear but cruel **boyfriend** recently.
School	Examination failed or grade dropped; Unable to enter/stay in school	I hate to go to **school** because my **teacher** always **blames** me.
Work	Laid off or fired from a job; Demotion and salary reduction	I **lost** my **job** in this economic recession a few months ago.
Social	Substantial conflicts with a friend; Difficulties in social activities	I **argued** with my best **friend** and was upset.

Table 1. Classification of negative life events.

<school, teacher, blame> in the example sentences in Table 1. Such language patterns are not necessarily composed of continuous words. Instead, they are usually composed of the words with long-distance dependencies, which cannot be easily captured by *n*-grams.

Therefore, the aim of this study is two-fold: (1) to automatically discover association language patterns from the sentences annotated with negative life events; and (2) to classify sentences with negative life events using the discovered patterns. To discover association language patterns, we incorporate the measure mutual information (MI) into a data mining algorithm, called *association pattern mining*, to incrementally derive frequently co-occurred words in sentences (Section 2). The discovered patterns are then combined with single words as features to train classifiers for negative life event classification (Section 3). Experimental results are presented in Section 4. Conclusions are finally drawn in Section 5.

2 Association Language Pattern Mining

The problem of language pattern acquisition can be converted into the problem of association pattern mining, where each sales transaction in a database can be considered as a sentence in the corpora, and each item in a transaction denotes a word in a sentence. An association language pattern is defined herein as a combination of multiple associated words, denoted by $< w_1,...,w_k >$. Thus, the task of association pattern mining is to mine the language patterns of frequently associated words from the training sentences. For this purpose, we adopt the Apriori algorithm (Agrawal and Srikant, 1994) and modified it slightly to fit our application. Its basic concept is to identify frequent word sets recursively, and then generate association language patterns from the frequent word sets. For simplicity, only the combinations of nouns and verbs are considered, and the length is restricted to at most 4 words, i.e., 2-word, 3-word and 4-word combinations. The detailed procedure is described as follows.

2.1 Find frequent word sets

A word set is frequent if it possesses a minimum support. The support of a word set is defined as the number of training sentences containing the word set. For instance, the support of a two-word set $\{ w_i , w_j \}$ denotes the number of training sentences containing the word pair (w_i , w_j). The frequent *k*-word sets are discovered from (*k*-1)-word sets. First, the support of each word, i.e., word frequency, in the training corpus is counted. The set of frequent one-word sets, denoted as L_1, is then generated by choosing the words with a minimum support level. To calculate L_k, the following two-step process is performed iteratively until no more frequent *k*-word sets are found.

- **Join step:** A set of candidate *k*-word sets, denoted as C_k, is first generated by merging frequent word sets of L_{k-1}, in which only the word sets whose first (k-2) words are identical can be merged.
- **Prune step:** The support of each candidate word set in C_k is then counted to determine which candidate word sets are frequent. Finally, the candidate word sets with a support count greater than or equal to the minimum support are considered to form L_k. The candidate word sets with a subset that is not frequent are eliminated. Figure 1 shows an example of generating L_k.

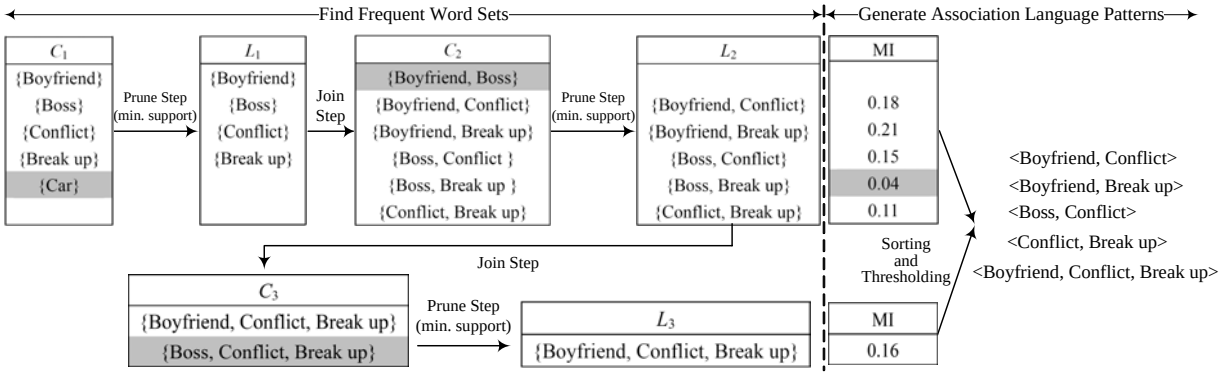


Figure 1. Example of generating association language patterns.

2.2 Generate association patterns from frequent word sets

Association language patterns can be generated via a confidence measure once the frequent word sets have been identified. The confidence of an association language pattern of *k* words is defined as the mutual information of the *k* words, as shown below.

$$Conf(< w_1,...w_k >) = MI(w_1,...w_k)$$
$$= P(w_1,...w_k)\log\frac{P(w_1,...w_k)}{\prod_{i=1}^{k} P(w_i)} \quad (1)$$

where $P(w_1,...w_k)$ denotes the probability of the *k* words co-occurring in a sentence in the training set, and $P(w_i)$ denotes the probability of a single word occurring in the training set. Accordingly, each frequent word set in L_k is assigned a mutual information score. In order to generate a set of association language patterns, all frequent word sets are sorted in the descending order of the mutual information scores. The minimum confidence (a threshold at percentage) is then applied to select top *N* percent frequent word sets as the resulting language patterns. This threshold is determined empirically by maximizing classification performance (Section 4). Figure 1 (right-hand side) shows an example of generating the association language patterns from L_k.

3 Sentence Classification

The classifiers used in this study include Support Vector Machine (SVM), C4.5, and Naïve Bayes (NB) classifier, which is provided by Weka Package (Witten and Frank, 2005). The feature set includes:

Bag-of-Words (BOW): Each single word in sentences.

Association language patterns (ALP): The top *N* percent association language patterns acquired in the previous section.

Ontology expansion (Onto): The top *N* percent association language patterns are expanded by mapping the constituent words into their synonyms. For example, the pattern <boss, conflict> can be expanded as <chief, conflict> since the words *boss* and *chief* are synonyms. Here we use the HowNet (http://www.keenage.com), a Chinese lexical ontology, for pattern expansion.

4 Experimental Results

Data set: A total of 2,856 sentences were collected from the Internet-based Self-assessment Program for Depression (ISP-D) database of the PsychPark (http://www.psychpark.org), a virtual psychiatric clinic, maintained by a group of volunteer professionals of Taiwan Association of Mental Health Informatics (Bai et al., 2001). Each sentence was then annotated by trained annotators with one of the five types of negative life events. Table 2 shows the break-down of the distribution of sentence types.

The data set was randomly split into a training set, a development set, and a test set with an 8:1:1 ratio. The training set was used for language pattern generation. The development set was used to optimize the threshold (Section 2.2) for the classifiers (SVM, C4.5 and NB). Each classifier was implemented using three different levels of features, namely BOW, BOW+ALP,

Sentence Type	% in Corpus
Family	28.8
Love	22.8
School	13.3
Work	14.3
Social	20.8

Table 2. Distribution of sentence types.

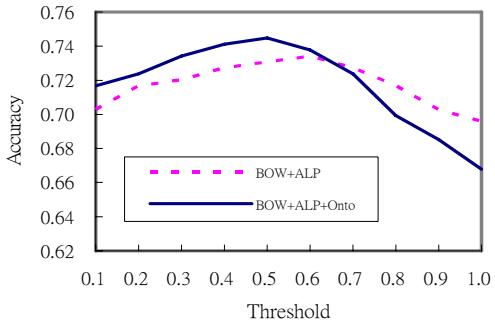


Figure 2. Threshold selection.

and BOW+ALP+Onto, to examine the effectiveness of association language patterns. The classification performance is measured by *accuracy*, i.e., the number of correctly classified sentences divided by the total number of test sentences.

4.1 Evaluation on threshold selection

Since not all discovered association language patterns contribute to the classification task, the threshold described in Section 2.2 is used to select top *N* percent patterns for classification. This experiment is to determine an optimal threshold for each involved classifier by maximizing its classification accuracy on the development set. Figure 2 shows the classification accuracy of NB against different threshold values.

When using association language patterns as features (BOW+ALP), the accuracy increased with increasing the threshold value up to 0.6, indicating that the top 60% discovered patterns contained more useful patterns for classification. By contrast, the accuracy decreased when the threshold value was above 0.6, indicating that the remaining 40% contained more noisy patterns that may increase the ambiguity in classification. When using the ontology expansion approach (BOW+ALP+Onto), both the number and diversity of discovered patterns are increased. Therefore, the accuracy was improved and the optimal accuracy was achieved at 0.5. However, the accuracy dropped significantly when the threshold value was above 0.5. This finding indicates that expansion on noisy patterns may produce more noisy patterns and thus decrease performance.

4.2 Results of classification performance

The results of each classifier were obtained from the test set using its own threshold optimized in the previous section. Table 3 shows the comparative results of different classifiers with different levels of features. The incorporation of association language patterns improved the accuracy of NB, C4.5, and SVM by 3.9%, 1.9%, and 2.2%, respectively, and achieved an average improvement of 2.7%. Additionally, the use of ontology expansion can further improve the performance by 1.6% in average. This finding indicates that association language patterns are significant features for negative life event classification.

	NB	C4.5	SVM
BOW	0.717	0.741	0.787
BOW+ALP	0.745	0.755	0.804
BOW+ALP+Onto	0.759	0.766	0.815

Table 3. Accuracy of classifiers on testing data.

5 Conclusion

This work has presented a framework that uses a data mining algorithm and ontology expansion method to acquire association language patterns for negative life event classification. The association language patterns can capture word relationships in sentences, thus yielding higher performance than the baseline system using single words alone. Future work will focus on devising a semi-supervised or unsupervised method for language pattern acquisition from web resources so as to reduce reliance on annotated corpora.

References

R. Agrawal and R. Srikant. 1994. Fast Algorithms for Mining Association Rules. In *Proc. Int'l Conf. Very Large Data Bases (VLDB)*, pages 487-499.

Y. M. Bai, C. C. Lin, J. Y. Chen, and W. C. Liu. 2001. Virtual Psychiatric Clinics. *American Journal of Psychiatry*, vol. 158, no. 7, pp. 1160-1161.

E. M. Brostedt and N. L. Pedersen. 2003. Stressful Life Events and Affective Illness. *Acta Psychiatrica Scandinavica*, vol. 107, pp. 208-215.

R. Chitturi and J. H.L. Hansen. 2008. Dialect Classification for online podcasts fusing Acoustic and Language based Structural and Semantic Information. In *Proc. of ACL-08*, pages 21-24.

A. Khoo, Y. Marom and D. Albrecht. 2006. Experiments with Sentence Classification. In *Proc. of Australasian Language Technology Workshop*, pages 18-25.

S. Li and C. Zong. 2008. Multi-domain Sentiment Classification. In *Proc. of ACL-08*, pages 257-260.

M. Naughton, N. Stokes, and J. Carthy. 2008. Investigating Statistical Techniques for Sentence-Level Event Classification. In *Proc. of COLING-08*, pages 617-624.

F. Sebastiani. 2002. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, vol. 34, no. 1, pp. 1-47.

I. H. Witten and E. Frank. 2005. Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition, Morgan Kaufmann, San Francisco.

Automatic Compilation of Travel Information from Automatically Identified Travel Blogs

Hidetsugu Nanba
Graduate School of Information Sciences, Hiroshima City University
nanba@hiroshima-cu.ac.jp

Haruka Taguma
School of Information Sciences, Hiroshima City University

Takahiro Ozaki
School of Information Sciences, Hiroshima City University

Daisuke Kobayashi
Graduate School of Information Sciences, Hiroshima City University
kobayashi@ls.info.hiroshima-cu.ac.jp

Aya Ishino
Graduate School of Information Sciences, Hiroshima City University
ishino@ls.info.hiroshima-cu.ac.jp

Toshiyuki Takezawa
Graduate School of Information Sciences, Hiroshima City University
takezawa@hiroshima-cu.ac.jp

Abstract

In this paper, we propose a method for compiling travel information automatically. For the compilation, we focus on travel blogs, which are defined as travel journals written by bloggers in diary form. We consider that travel blogs are a useful information source for obtaining travel information, because many bloggers' travel experiences are written in this form. Therefore, we identified travel blogs in a blog database and extracted travel information from them. We have confirmed the effectiveness of our method by experiment. For the identification of travel blogs, we obtained scores of 38.1% for Recall and 86.7% for Precision. In the extraction of travel information from travel blogs, we obtained 74.0% for Precision at the top 100 extracted local products, thereby confirming that travel blogs are a useful source of travel information.

1 Introduction

Travel guidebooks and portal sites provided by tour companies and governmental tourist boards are useful sources of information about travel. However, it is costly and time consuming to compile travel information for all tourist spots and to keep them up to date manually. Therefore we have studied the automatic compilation of travel information.

For the compilation, we focused on travel blogs, which are defined as travel journals written by bloggers in diary form. Travel blogs are considered a useful information source for obtaining travel information, because many bloggers' travel experiences are written in this form. Therefore, we identified travel blogs in a blog database, and extracted travel information from them.

Travel information in travel blogs is also useful for recommending information that is matched to the each traveler. Recently, several methods that identify bloggers' attributes such as residential area (Yasuda *et al.*, 2006), gender, and age (Ikeda *et al.*, 2008, Schler *et al.*, 2006), have been proposed. By combining this research with travel information extracted from travel blogs, it is possible to recommend a local product that is popular among females, for example, or a travel spot, where young people often visit.

The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 describes our method. To investigate the effectiveness of our method, we conducted some experiments, and Section 4 reports the experimental results. We present some conclusions in Section 5.

2 Related Work

Both 'www.travelblog.org' and 'travel.blogmura.com' are portal sites for travel blogs. At these sites, travel blogs are manually registered by bloggers themselves, and the blogs are classified by their destinations. However, there are many more travel blogs in the blogos-

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 205–208,
Suntec, Singapore, 4 August 2009.

phere. Aiming to construct an exhaustive database of travel blogs, we have studied the automatic identification of travel blogs.

GeoCLEF[1] is the cross-language geographic retrieval track run as part of the Cross Language Evaluation Forum (CLEF), and has been operating since 2005 (Gey *et al.*, 2005). The goal of this task was to retrieve news articles relevant to particular aspects of geographic information, such as 'wine regions around the rivers in Europe'. In our work, we focused on travel blogs instead of news articles, because bloggers' travel experiences tend to be written in travel blogs.

3 Automatic Compilation of Travel Information

The task of compiling travel information is divided into two steps: (1) identification of travel blogs and (2) extraction of travel information from them. We explain these steps in Sections 3.1 and 3.2.

3.1 Identification of Travel Blogs

Blog entries that contain cue phrases, such as 'travel', 'sightseeing', or 'tour', have a high degree of probability of being travel blogs. However, not every travel blog contains such cue phrases. For example, if a blogger writes his/her journey to Norway in multiple blog entries, it might state 'We traveled to Norway' in the first entry, while only writing 'We ate wild sheep!' in the second entry. In this case, because the second entry does not contain any expressions related to travel, it is difficult to identify that the second entry is a travel blog. Therefore, we focus not only on each entry but also on its surrounding entries for the identification of travel blogs.

We formulated the identification of travel blogs as a sequence-labeling problem, and solved it using machine learning. For the machine learning method, we examined the Conditional Random Fields (CRF) method, whose empirical success has been reported recently in the field of natural language processing. The CRF-based method identifies the class of each entry. Features and tags are given in the CRF method as follows: (1) the k tags occur before a target entry, (2) k features occur before a target entry, and (3) k features follow a target entry (see Figure 1). We used the value of k=4, which was determined in a pilot study. Here, we used the following features for machine learning: whether an entry contains each 416 cue phrase, such as '旅行 (travel)', 'ツアー (tour)', and '出発 (departure)', and the number of location names in each entry[2].

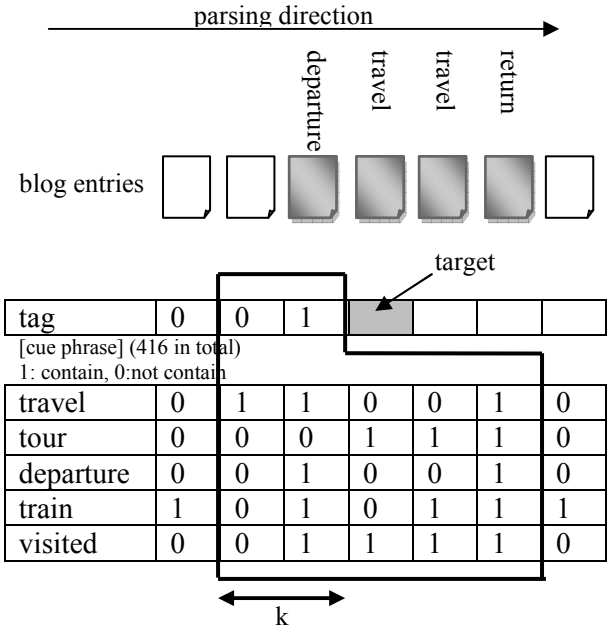


Figure 1: Features and tags given to the CRF

3.2 Extraction of Travel Information from Blogs

We extracted pairs comprising a location name and a local product from travel blogs, which were identified in the previous step. For the efficient extraction of travel information, we employed a bootstrapping method. Firstly, we prepared 482 location-name/and local-product pairs as seeds for the bootstrapping. These pairs were obtained automatically from a 'Web Japanese N-gram' database[3] provided by Google, Inc. The database comprises N-grams (N=1–7) extracted from 20 billion of Japanese sentences on the web. We applied a pattern '[地名]名物「[名物]」' ([slot of 'location name'] local product 「[slot of 'local product name']」) to the database, and extracted location names and local products from each corresponding slot, thereby obtaining the 482 pairs.

Secondly, we applied a machine learning-based information extraction technique to the travel blogs identified in the previous step, and obtained new pairs. In this step, we prepared

[1] http://ir.shef.ac.uk/geoclef/

[2] We used CaboCha software for the identification of locations.
http://chasen.org/~taku/software/cabocha/

[3] http://www.gsk.or.jp/catalog/GSK2007-C/catalog.html

training data for the machine learning in the following three steps.

1. Select 200 sentences that contain both a location name and a local product from the 482 pairs. Then automatically create 200 tagged sentences, to which 'location' and 'product' tags are assigned.
2. Prepare another 200 sentences that contain only a location name.[4] Then create 200 tagged sentences, to which the 'location' tag is assigned.
3. Apply machine learning to the 400 tagged sentences, and obtain a system that automatically annotates 'location' and 'product' tags to given sentences.

As a machine learning method, we used the CRF. In the same way as in the previous step, the CRF-based method identifies the class of each word in a given sentence. Features and tags are given in the CRF method as follows: (1) the k tags occur before a target word, (2) k features occur before a target word, and (3) k features follow a target word. We used the value of k=2, which was determined in a pilot study. We use the following six features for machine learning.

- A word.
- Its part of speech[5].
- Whether the word is a quotation mark.
- Whether the word is a cue word, such as '名物', '名産', '特産' (local product), '銘菓' (famous confection), or '土産' (souvenir).
- Whether the word is a surface case.
- Whether the word is frequently used in the names of local products or souvenirs, such as 'cake' or 'noodle'.

4 Experiments

We conducted two experiments: (1) identification of travel blogs, and (2) extraction of travel information from blogs. We reported on them in Sections 4.1 and 4.2.

4.1 Identification of Travel Blogs

Data sets and experimental settings

We randomly selected 4,914 blog entries written by 317 authors from about 1,100,000 entries written in Japanese. Then we manually identified travel blogs in 4,914 entries. As a result, 420 entries were identified as travel blogs. Then we performed a four-fold cross-validation test. For the machine-learning package, we used CRF++[6] software. For evaluation measures, we used Recall and Precision scores.

Alternatives

In order to confirm the validity of our sequence labeling-based approach, we also examined another method, which identifies travel blogs using features in each blog entry only (without using features in its surrounding entries).

Results and discussions

Table 1 shows the experimental results. As shown in the table, our method improved the Precision value by 26.2%, while decreasing the Recall value by 13.0%. In our research, Precision is more important than Recall, because low Precision in this step causes low Precision in the next step.

	Recall	Precision
our method	38.1	86.7
baseline method	51.1	60.5

Table 1: Identification of travel blogs

Our method could not identify 266 of the travel blogs. We randomly selected 50 entries from these 266, and analysed the errors. Among the 50 errors, 25 cases (50%) were caused by the lack of cue phrases. For the machine learning, we used manually selected cue phrases. To increase the number of cue phrases, a statistical approach will be required. For example, applying n-grams to automatically identified travel blogs is one such approach. Among the 50 errors, 5 entries (10%) were too short (fewer than four sentences) to be identified by our method.

Our method mistakenly identified 26 entries as travel blogs. A typical error is that bloggers wrote non-travel entries among a series of travel blogs. In this case, the non-travel entries were identified as travel blogs.

4.2 Extraction of Travel Information from Blogs

Data sets and experimental settings

To confirm that travel blogs are a useful information source for the extraction of travel information, we extracted travel information using the following three information sources.

[4] In our pilot study, we did not use these negative cases in machine learning at first, and obtained low precision values, because our system attempted to extract local products from all sentences containing location names in travel blogs.

[5] In this step, we also identified location names automatically using the CaboCha software.

[6] http://www.chasen.org/~taku/software/CRF++/

- **Travel blogs (our method)**: 80,000 sentences in 17,268 travel blogs, which were automatically identified from 1,100,000 entries using the method described in Section 3.1.
- **Generic blogs**: 80,000 sentences from 1,100,000 blog entries.
- **Generic webs**: 80,000 sentences from 470M web sentences (Kawahara and Kurohashi, 2006).

We extracted travel information (location-name/local-product pairs) from each information source, and ranked them by their frequencies.

Evaluation

We used the Precision value for the top-ranked travel information defined by the following equation as the evaluation measure. We calculated Precision values from the top 5 to the top 100 at intervals of 5.

$$\text{Precision} = \frac{\text{The number of correctly extracted location-name / local-product pairs}}{\text{The number of extracted location-name / local-product pairs}}$$

Results and discussions

Figure 2 shows the experimental results. As shown in the figure, the generic blog method obtained higher Precision values than the generic web method, especially at higher ranks. Our method (travel blog) was much better than the generic blog method, which indicates that travel blogs are a useful information source for the extraction of travel information.

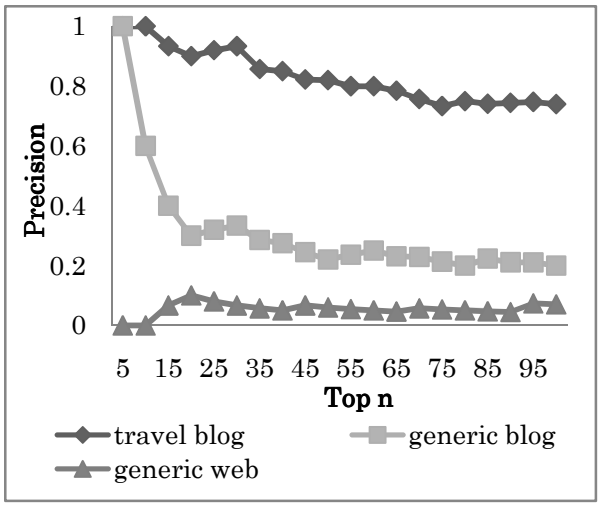


Figure 2: Precision values at top n for the extraction of travel information

Table 2 shows the number of local products, which were not contained in a list of products from the Google N-gram database. As shown in the table, 41 local products were newly extracted from travel blogs, while 15 and 7 were extracted from generic blogs and generic webs, respectively. These results also indicate the effectiveness of travel blogs as a source for travel information.

A typical error among the top 100 results for our method was that store names were mistakenly extracted. Here, most of these stores sell local products. To ameliorate this problem, extraction of pairs of local products and the stores that sell them is also required.

travel blog (our method)	41
generic blog	15
generic web	7

Table 2: The number of local products that each method newly extracted

5 Conclusion

In this paper, we proposed a method for identifying travel blogs from a blog database, and extracting travel information from them. In the identification of travel blogs, we obtained of 38.1% for Recall and 86.7% for Precision. In the extraction of travel information from travel blogs, we obtained 74.0% for Precision with the top 100 extracted local products.

References

Fredric C. Gey, Ray R. Larson, Mark Sanderson, Hideo Joho, Paul Clough, and Vivien Petras. 2005. GeoCLEF: The CLEF 2005 Cross-Language Geographic Information Retrieval Track Overview. *Lecture Notes in Computer Science*, LNCS4022, pp.908-919.

Daisuke Ikeda, Hiroya Takamura, and Manabu Okumura. 2008. Semi-Supervised Learning for Blog Classification. *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp.1156-1161.

Daisuke Kawahara and Sadao Kurohashi. 2006. A Fully-Lexicalized Probabilistic Model for Japanese Syntactic and Case Structure Analysis. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp.176-183.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James Pennebaker. 2006. Effects of age and gender on blogging. *Proceedings of AAAI Symposium on Computational Approaches for Analyzing Weblogs*, pp.199-205.

Norihito Yasuda, Tsutomu Hirao, Jun Suzuki, and Hideki Isozaki. 2006. Identifying bloggers' residential areas. *Proceedings of AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*, pp.231-236.

Play the Language: Play Coreference

Barbora Hladká and **Jiří Mírovský** and **Pavel Schlesinger**
Charles University in Prague
Institute of Formal and Applied Linguistics
e-mail: {hladka, mirovsky, schlesinger@ufal.mff.cuni.cz}

Abstract

We propose the PlayCoref game, whose purpose is to obtain substantial amount of text data with the coreference annotation. We provide a description of the game design that covers the strategy, the instructions for the players, the input texts selection and preparation, and the score evaluation.

1 Introduction

A collection of high quality data is resource-demanding regardless of the area of research and type of the data. This fact has encouraged a formulation of an alternative way of data collection, "Games With a Purpose" methodology (GWAP), (van Ahn and Dabbish, 2008). The GWAP methodology exploits the capacity of Internet users who like to play on-line games. The on-line games are being designed to generate data for applications that either have not been implemented yet, or have already been implemented with a performance lower than human. Moreover, the players work simply by playing the game - the data are generated as a by-product of the game. If the game is enjoyable, it brings human resources and saves financial resources. The game popularity brings more game sessions and thus more annotated data.

The GWAP methodology was formulated in parallel with design and implementation of the on-line games with images (van Ahn and Dabbish, 2004) and subsequently with tunes (Law et al., 2007),[1] in which the players try to agree on a caption of the image/tune. The popularity of the games is enormous so the authors have succeeded in the basic requirement that the annotation is generated in a substantial amount. Then the *Onto games* appeared (Siorpaes and Hepp, 2008), bringing a new type of input data to GWAP, namely video and text.[2]

The situation with text seems to be slightly different. One has to read a text in order to identify its topics, which takes more time than observing images, and the longer text, the worse. Since the game must be of a dynamic character, it is unimaginable that the players will spend minutes reading an input text. Therefore, the text must be opened to the players 'part' by 'part'.

So far, besides the Onto games, two more games with texts have been designed: *What did Shannon say?*[3], the goal of which is to help the speech recognizer with difficult-to-recognize words, and *Phrase Detectives*[4] (Kruschwitz, Chamberlain, Poesio, 2009), the goal of which is to identify relationships between words and phrases in a text.

Motivated by the GWAP portal, the LGame portal[5] has been established. Seven key properties that any game on the LGame portal will satisfy were formulated – see Table 1.

The LGame portal has been opened with the *Shannon game*, a game of intentionally hidden words in the sentence, where players guess them, and the *Place the Space* game, a game of word segmentation.

Within a systematic framework established at the LGame portal, the games *PlayCoref, PlayNE, PlayDoc* devoted to the linguistic phenomena dealing with the contents of documents, namely coreference, named-entitites, and document labels, respectively, are being designed in parallel but implemented subsequently since the GWAPs are open-ended stories the success of which is hard to estimate in advance. These games are designed for Czech and English by default. However, the game rules are language independent.

[1] www.gwap.org

[2] www.ontogame.org

[3] lingo.clsp.jhu.edushannongame.html

[4] www.phrasedetectives.org

[5] www.lgame.cz

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 209–212,
Suntec, Singapore, 4 August 2009.

1. During the game, the data are collected for the natural language processing tasks that computers cannot solve at all or not well enough.
2. Playing the game only requires a basic knowledge of the grammar of the language of the game. No extra linguistic knowledge is required.
3. The game rules are designed independently of the language of the game.
4. The game is designed for Czech and English by default.
5. During the game, the players have at least a general idea of what their opponent(s) do.
6. The game is designed for at least two players (also a computer can be an opponent).
7. The game offers several levels of difficulty (to fit a vast range of players).

Table 1: Key properties of the games on the LGame portal.

We have decided to implement the PlayCoref first. Coreference crosses the sentence boundaries and playing coreference offers a great opportunity to test players' willingness to read a text part by part, e.g. sentence by sentence. In this paper, we discuss various aspects of the PlayCoref design.

2 Coreference

Coreference occurs when several referring expressions in a text refer to the same entity (e.g. person, thing, reality). A *coreferential pair* is marked between subsequent pairs of the referring expressions. A sequence of coreferential pairs referring to the same entity in a text forms a *coreference chain*.

Various projects on the coreference annotation by linguists are running. We mention two of them – the Prague Dependency Treebank 2.0 and the coreference task for the sixth Message Understanding Conference.

Prague Dependency Treebank 2.0 (PDT 2.0)[6] is the only corpus establishing the coreference annotation on a layer of meaning, so-called tectogrammatical layer (t-layer). The annotation includes grammatical and textual coreference. Extended textual coreference (covering additional categories) is being annotated in PDT 2.0 in an ongoing project (Nedoluzhko, 2007).

Sixth Message Understanding Conference – the coreference task (MUC-6)[7] operates on a surface layer. The coreferential pairs are marked between pairs of the categories nouns, noun phrases, and pronouns.

[6] ufal.mff.cuni.cz/pdt2.0

[7] cs.nyu.edu/faculty/grishman/muc6.html

3 The PlayCoref Game

Motivation The PDT 2.0 coreference annotation (including the annotation scheme design, training of the annotators, technical and linguistic support, and annotation corrections) spanned the period from summer 2002 till autumn 2004. Each of two annotators annotated one half out of 3,165 documents. We are aware that coreferential pairs marked in the PlayCoref sessions may differ from the PDT 2.0 coreference annotation. However, the following estimates reinforce our motivation to use the GWAP technology on texts: assuming that (1) the PlayCoref is designed as a two-player game, (2) at least one document is being present in each session, (3) the session lasts up to 5 minutes and (4) the players play half an hour a day, then at least 6 documents will be processed a day by two players. This means that 3,165 documents will be annotated by two players in 528 days, by eight players in 132 days, by 32 players in 33 days etc., and by 128 players in 9 days.

Strategy The game is designed for two players. The game starts with several first sentences of the document displayed in the players' sentence window. According to the restrictions put on the members of the coreferential pairs, parts of the text are unlocked while the other parts are locked. Only unlocked parts of the text are allowed to become a member of the coreferential pair. In our case, only nouns and selected pronouns are unlocked.[8] In Table 2, we provide a list of the locked pronoun's sub-part-of-speech classes (as designed in the Czech positional tag system). Pronouns of the other sub-part-of-speech classes are unlocked. The selection of the locked pronoun's sub-part-of-speech classes is based on the fact that some types of pronouns usually corefer with parts of the text larger than one word. This type of coreference cannot be annotated without a linguistic knowledge and without training. Therefore it must be omitted for the purposes of the PlayCoref game.

The players mark coreferential pairs between the unlocked words in the text (no phrases are allowed). They mark the coreferential pairs as undirected links.[9] After the session, the coreference

[8] A tagging procedure is used to get the part-of-speech classes of the words.

[9] This strategy differs from the general conception of coreference being understood as either the anaphoric or cataphoric relation depending on "direction" of the link in the text. We believe that the players will benefit from this sim-

Locked pronouns: subPOS and its description	
D	Demonstrative (”ten”, ”onen”, ..., lit. ”this”, ”that”, ”that”, ... ”over there”, ...)
E	Relative ”což” (corresponding to English which in subordinate clauses referring to a part of the preceding text)
L	Indefinite ”všechen”, ”sám” (lit. ”all”, ”alone”)
O	”svůj”, ”nesvůj”, ”tentam” alone (lit. ”own self”, ”not-in-mood”, ”gone”)
Q	Relative/interrogative ”co”, ”copak”, ”cožpak” (lit. ”what”, ”isn’t-it-true-that”)
W	Negative (”nic”, ”nikdo”, ”nijaký”, ”žádný”, ..., lit. ”nothing”, ”nobody”, ”not-worth-mentioning”, ”no”/”none”)
Y	Relative/interrogative ”co” as an enclitic (after a preposition) (”oč”, ”nač”, ”zač”, lit. ”about what”, ”on”/”onto” ”what”, ”after”/”for what”)
Z	Indefinite (”nějaký”, ”některý”, ”číkoli”, ”cosi”, ..., lit. ”some”, ”some”, ”anybody’s”, ”something”)

Table 2: List of the pronoun’s sub-part-of-speech classes in the Czech positional tag system locked for the PlayCoref.

chains are automatically reconstructed from the coreferential pairs marked.

During the session, the number of words the opponent has linked into the coreferential pairs is displayed to the player. The number of sentences with at least one coreferential pair marked by the opponent is displayed to the player as well. Revealing more information about the opponent’s actions would affect the independency of the players’ decisions.

If the player finishes pairing all the related words in a visible part of the document (visible to him), he asks for the next sentence of the document. It appears at the bottom of his sentence window. The player can remove pairs created before at any time and can make new pairs in the sentences read so far. The session goes on this way until the end of the session time.

Instructions for the Players Instructions for the players must be as comprehensible and concise as possible. To mark a coreferential pair, no linguistic knowledge is required. It is all about the text comprehension ability.

Input Texts In the first stage of the project, documents from PDT 2.0 and MUC-6 will be used in the sessions, so that the quality of the game data can be evaluated against the manual coreference annotation.

Since the PDT 2.0 coreference annotation operates on the tectogrammatical layer and PlayCoref on the surface layer, the coreferential pairs of the t-layer must be projected to the surface first. The basic steps of the projection are depicted in Figure 1. Going from the t-layer, some of the coreferential pairs get lost because their members do not have their counterparts on surface.[10] From the remaining coreferential pairs, those between nouns and unlocked pronouns are selected. In the final game documents, the difference between the grammatical, textual and extended textual coreference is omitted, because the players will not be asked to distinguish them. Table 3 shows the number of coreferential pairs in various stages of the projection.

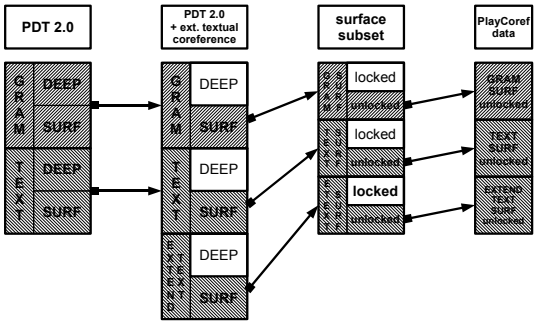


Figure 1: Projection of the PDT coreference annotation to the surface layer. The first step depicts the annotation of the extended textual coreference. Pairs that have no surface counterparts are marked *DEEP*, pairs with surface counterparts are marked *SURF*. Pairs suitable for the game are marked *unlocked*.

Data from the coreference task on the sixth Message Understanding Conference can be used in a much more straightforward way. Coreference is annotated on the surface and no projection is needed. The links with noun phrases are disregarded.

	PDT 2.0	PDT 2.0 + ext.	surface subset	PlayCoref
# coref. pairs	45	96	70	33

Table 3: Number of coreferential pairs (in thousands) in various stages of projection. Counts in the second, third and fourth columns are extrapolated on the basis of data annotated so far, which is about 200 thousand word tokens in 12 thousand sentences (out of 833 thousand tokens in 49 thousand sentences in PDT 2.0). Type of the coreferential pairs, either grammatical or textual one, is not distinguished.

Scoring The players get points for their coreferential pairs according to the equation $pts_A = w_1 * ICA(A, acr) + w_2 * ICA(A, B)$ where A and B are the players, acr is an automatic coreference resolution procedure, weights $0 \leq w_1, w_2 \leq 1$, $w_1, w_2 \in R$ are set empirically, and ICA stands for the inter-coder agreement that we can simultaneously express either by the F-measure or Krippen-

plification and that the quality of the game data will not be decreased.

[10] Czech is a ’pro-drop’ language, in which the subject pronoun on ’he’ has a zero form (also in feminine, plural, etc.).

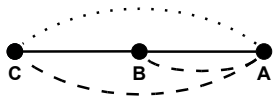


Figure 2: Player '1' pairs (A,C) – the dotted curve; player '2' pairs (A,B) and (B,C) – the solid lines; player '3' pairs (A,B) and (A,C) – the dashed curves. Although players '1' and '2' do not agree on the coreferential pairs at all, '1' and '3' agree only on (A,C) and '2' and '3' agree only on (A,B), for the purposes of the coreference chains reconstruction, the players' agreement is higher: players '1' and '2' agree on two members of the coreferential chain: A and C, players '1' and '3' agree on A and C as well, and players '2' and '3' achieved agreement even on all three members: A, B, and C.

dorff's α (Artstein and Poesio, 2008). The score is calculated at the end of the session and no running score is being presented during the session. Otherwise, the players might adjust their decisions according to the changes in the score. Obviously, it is undesirable.

Assigning a score to the players deals with the coreferential pairs. However, motivated by (Passonneau, 2004) and others, the evaluation handles the coreferential pairs in a way demonstrated in Figure 2.

PlayCoref vs. PhraseDetectives At least to our knowledge, there are no other GWAPs dealing with the relationship among words in a text like PhraseDetectives and PlayCoref. Nevertheless, there are many differences between these two games – the main ones are enumerated in Table 4.

PlayCoref	***PhraseDetectives***
detection of coreference chains	anaphora resolution
two-player game	one-player game
a document presented sentence by sentence	a paragraph presented at once
–	checking the pairs marked in the previous sessions
pairing not restricted to the position in the text	the closest antecedent
simple instructions	players training
scoring with respect to the automatic coreference resolution and to the opponent's pairs	scoring with respect to the players that play with the same document before
coreferential pairs correction	no corrections allowed

Table 4: PlayCoref vs. PhraseDetectives.

4 Conclusion

We propose the PlayCoref game, a concept of a GWAP with texts that aims at getting the documents with the coreference annotation in substantially larger volume than can be obtained from experts. In the proposed game, we introduce coreference to the players in a way that no linguistic knowledge is required from them. We present the game rules design, the preparation of the game documents and the evaluation of the players' score. A short comparison with a similar project is also provided.

Acknowledgments

We gratefully acknowledge the support of the Czech Ministry of Education (grants MSM-0021620838 and LC536), the Czech Grant Agency (grant 405/09/0729), and the Grant Agency of Charles University in Prague (project GAUK 138309).

References

Ron Artstein, Massimo Poesio. 2008. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, December 2008, vol. 34, no. 4, pp. 555–596.

Udo Kruschwitz, Jon Chamberlain, Massimo Poesio. 2009. (Linguistic) Science Through Web Collaboration in the ANAWIKI project. In *Proceedings of the WebSci'09: Society On-Line*, Athens, Greece, in press.

Lucie Kučová, Eva Hajičová. 2005. Coreferential Relations in the Prague Dependency Treebank. In *Proceedings of the 5th International Conference on Discourse Anaphora and Anaphor Resolution*, San Miguel, Azores, pp. 97–102.

Edith. L. M. Law et al. 2007. Tagatune: A game for music and sound annotation. In *Proceedings of the Music Information Retrieval Conference*, Austrian Computer Soc., pp. 361–364.

Anna Nedoluzhko. 2007. Zpráva k anotování rozšířené textové koreference a bridging vztahů v Pražském závoslostním korpusu (Annotating extended coreference and bridging relations in PDT). Technical Report, UFAL, MFF UK, Prague, Czech Republic.

Rebecca J. Passonneau. 2004. Computing Reliability for Coreference. *Proceedings of LREC*, vol. 4, pp. 1503–1506, Lisbon.

Katharina Siorpaes and Martin Hepp. 2008. Games with a purpose for the Semantic Web. *IEEE Intelligent Systems Vol. 23, number 3*, pp. 50–60.

Luis van Ahn and Laura Dabbish. 2004. Labelling images with a computer game. In *Proceedings of the SIGHI Conference on Human Factors in Computing Systems, ACM Press*, New York, pp. 319–326.

Luis van Ahn and Laura Dabbish. 2008. Designing Games with a Purpose. *Communications of the ACM*, vol. 51, No. 8, pp. 58–67.

Chinese Term Extraction Using Different Types of Relevance

Yuhang Yang[1], Tiejun Zhao[1], Qin Lu[2], Dequan Zheng[1] and Hao Yu[1]
[1]School of Computer Science and Technology,
Harbin Institute of Technology, Harbin 150001, China
{yhyang,tjzhao,dqzheng,yu}@mtlab.hit.edu.cn
[2]Department of Computing,
The Hong Kong Polytechnic University, Hong Kong, China
csluqin@comp.polyu.edu.hk

Abstract

This paper presents a new term extraction approach using relevance between term candidates calculated by a link analysis based method. Different types of relevance are used separately or jointly for term verification. The proposed approach requires no prior domain knowledge and no adaptation for new domains. Consequently, the method can be used in any domain corpus and it is especially useful for resource-limited domains. Evaluations conducted on two different domains for Chinese term extraction show significant improvements over existing techniques and also verify the efficiency and relative domain independent nature of the approach.

1 Introduction

Terms are the lexical units to represent the most fundamental knowledge of a domain. Term extraction is an essential task in domain knowledge acquisition which can be used for lexicon update, domain ontology construction, etc. Term extraction involves two steps. The first step extracts candidates by unithood calculation to qualify a string as a valid term. The second step verifies them through termhood measures (Kageura and Umino, 1996) to validate their domain specificity.

Many previous studies are conducted on term candidate extraction. Other tasks such as named entity recognition, meaningful word extraction and unknown word detection, use techniques similar to that for term candidate extraction. But, their focuses are not on domain specificity. This study focuses on the verification of candidates by termhood calculation.

Relevance between term candidates and documents is the most popular feature used for term verification such as *TF-IDF* (Salton and McGill, 1983; Frank, 1999) and *Inter-Domain Entropy* (Chang, 2005), which are all based on the hypothesis that "if a candidate occurs frequently in a few documents of a domain, it is likely a term". Limited distribution information of term candidates in different documents often limits the ability of such algorithms to distinguish terms from non-terms. There are also attempts to use prior domain specific knowledge and annotated corpora for term verification. *TV_ConSem* (Ji and Lu, 2007) calculates the percentage of context words in a domain lexicon using both frequency information and semantic information. However, this technique requires a domain lexicon whose size and quality have great impact on the performance of the algorithm. Some supervised learning approaches have been applied to protein/gene name recognition (Zhou et al., 2005) and Chinese new word identification (Li et al., 2004) using *SVM* classifiers (Vapnik, 1995) which also require large domain corpora and annotations. The latest work by Yang (2008) applied the relevance between term candidates and sentences by using the link analysis approach based on the *HITS* algorithm to achieve better performance.

In this work, a new feature on the relevance between different term candidates is integrated with other features to validate their domain specificity. The relevance between candidate terms may be useful to identify domain specific terms based on two assumptions. First, terms are more likely to occur with other terms in order to express domain information. Second, term candidates extracted from domain corpora are likely

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 213–216,
Suntec, Singapore, 4 August 2009.

to be domain specific. Previous work by (e.g. Ji and Lu, 2007) uses similar information by comparing the context to an existing large domain lexicon. In this study, the relevance between term candidates are iteratively calculated by graphs using link analysis algorithm to avoid the dependency on prior domain knowledge.

The rest of the paper is organized as follows. Section 2 describes the proposed algorithms. Section 3 explains the experiments and the performance evaluation. Section 4 concludes and presents the future plans.

2 Methodology

This study assumes the availability of term candidates since the focus is on term verification by termhood calculation. Three types of relevance are first calculated including (1) the term candidate relevance, *CC*; (2) the candidate to sentence relevance, *CS*; and the candidates to document relevance, *CD*. Terms are then verified by using different types of relevance.

2.1 Relevance between Term Candidates

Based on the assumptions that term candidates are likely to be used together in order to represent a particular domain concept, relevance of term candidates can be represented by graphs in a domain corpus. In this study, *CC* is defined as their co-occurrence in the same sentence of the domain corpus. For each document, a graph of term candidates is first constructed. In the graph, a node is a term candidate. If two term candidates TC_1 and TC_2 occur in the same sentence, two directional links between TC_1 to TC_2 are given to indicate their mutually related. Candidates with overlapped substrings are not removed which means long terms can be linked to their components if the components are also candidates.

After graph construction, the term candidate relevance, *CC*, is then iteratively calculated using the PageRank algorithm (Page et al. 1998) originally proposed for information retrieval. PageRank assumes that the more a node is connected to other nodes, it is more likely to be a salient node. The algorithm assigns the significance score to each node according to the number of nodes linking to it as well as the significance of the nodes. The PageRank calculation *PR* of a node *A* is shown as follows:

$$PR(A) = (1-d) + d\left(\frac{PR(B_1)}{C(B_1)} + \frac{PR(B_2)}{C(B_2)} + \ldots + \frac{PR(B_t)}{C(B_t)}\right) \quad (1)$$

where $B_1, B_2, \ldots, B_t$ are all nodes linked to node A; $C(B_i)$ is the number of outgoing links from node B_i; d is the factor to avoid loop trap in the graphic structure. d is set to 0.85 as suggested in (Page et al., 1998). Initially, all *PR* weights are set to 1. The weight score of each node are obtained by (1), iteratively. The significance of each term candidate in the domain specific corpus is then derived based on the significance of other candidates it co-occurred with. The *CC* weight of term candidate TC_i is given by its *PR* value after k iterations, a parameter to be determined experimentally.

2.2 Relevance between Term Candidates and Sentences

A domain specific term is more likely to be contained in domain relevant sentences. Relevance between term candidate and sentences, referred to as *CS*, is calculated using the *TV_HITS* (Term Verification – HITS) algorithm proposed in (Yang et al., 2008) based on Hyperlink-Induced Topic Search (*HITS*) algorithm (Kleinberg, 1997). In *TV_HITS*, a good hub in the domain corpus is a sentence that contains many good authorities; a good authority is a term candidate that is contained in many good hubs.

In *TV_HITS*, a node p can either be a sentence or a term candidate. If a term candidate *TC* is contained in a sentence *Sen* of the domain corpus, there is a directional link from *Sen* to *TC*. *TV_HITS* then makes use of the relationship between candidates and sentences via an iterative process to update *CS* weight for each *TC*.

Let $V^A(w(p_1)^A, w(p_2)^A, \ldots, w(p_n)^A)$ denote the authority vector and $V^H(w(p_1)^H, w(p_2)^H, \ldots, w(p_n)^H)$ denote the hub vector. V^A and V^H are initialized to (1, 1,…, 1). Given weights V^A and V^H with a directional link $p \rightarrow q$, $w(q)^A$ and $w(p)^H$ are updated by using the *I* operation(an in-pointer to a node) and the *O* operation(an out-pointer to a node) shown as follows. The *CS* weight of term candidate TC_i is given by its $w(q)^A$ value after iteration.

$$I \text{ operation: } w(q)^A = \sum_{p \rightarrow q \in E} w(p)^H \quad (2)$$

$$O \text{ operation: } w(p)^H = \sum_{p \rightarrow q \in E} w(q)^A \quad (3)$$

2.3 Relevance between Term Candidates and Documents

The relevance between term candidates and documents is used in many term extraction algo-

rithms. The relevance is measured by the *TF-IDF* value according to the following equations:

$$TFIDF(TC_i) = TF(TC_i) \cdot IDF(TC_i) \quad (4)$$

$$IDF(TC_i) = \log(\frac{|D|}{DF(TC_i)}) \quad (5)$$

where $TF(TC_i)$ is the number of times term candidate TC_i occurs in the domain corpus, $DF(TC_i)$ is the number of documents in which TC_i occurs at least once, $|D|$ is the total number of documents in the corpus, $IDF(TC_i)$ is the inverse document frequency which can be calculated from the document frequency.

2.4 Combination of Relevance

To evaluate the effective of the different types of relevance, they are combined in different ways in the evaluation. Term candidates are then ranked according to the corresponding termhood values *Th(TC)* and the top ranked candidates are considered terms.

For each document D_j in the domain corpus where a term candidate TC_i occurs, there is CC_{ij} weight and a CS_{ij} weight. When features *CC* and *CS* are used separately, termhood $Th_{CC}(TC_i)$ and $Th_{CS}(TC_i)$ are calculated by averaging CC_{ij} and CS_{ij}, respectively. Termhood of different combinations are given in formula (6) to (9). $R(TC_i)$ denotes the ranking position of TC_i.

$$Th_{CC+CS}(TC_i) = \frac{1}{R_{CC}(TC_i)} + \frac{1}{R_{CS}(TC_i)} \quad (6)$$

$$Th_{CC+CD}(TC_i) = (\sum_j CC_{ij}) \log(\frac{|D|}{DF_C}) \quad (7)$$

$$Th_{CS+CD}(TC_i) = (\sum_j CS_{ij}) \log(\frac{|D|}{DF_C}) \quad (8)$$

$$Th_{CC+CS+CD}(TC_i) = \frac{1}{R_{CC+CD}(TC_i)} + \frac{1}{R_{CS+CD}(TC_i)} \quad (9)$$

3 Performance Evaluation

3.1 Data Preparation

To evaluate the performance of the proposed relevance measures for Chinese in different domains, experiments are conducted on two separate domain corpora $Corpus_{IT}$ and $Corpus_{Legal}$., respectively. $Corpus_{IT}$ includes academic papers of 6.64M in size from Chinese IT journals between 1998 and 2000. $Corpus_{Legal}$ includes the complete set of official Chinese constitutional law articles and Economics/Finance law articles of 1.04M in size (http://www.law-lib.com/).

For comparison to previous work, all term candidates are extracted from the same domain corpora using the delimiter based algorithm *TCE_DI* (Term Candidate Extraction – Delimiter Identification) which is efficient according to (Yang et al., 2008). In *TCE_DI*, term delimiters are identified first. Words between delimiters are then taken as term candidates.

The performances are evaluated in terms of precision (P), recall (R) and *F*-value (F). Since the corpora are relatively large, sampling is used for evaluation based on fixed interval of 1 in each 10 ranked results. The verification of all the sampled data is carried out manually by two experts independently. To evaluate the recall, a set of correct terms which are manually verified from the extracted terms by different methods is constructed as the standard answer. The answer set is certainly not complete. But it is useful as a performance indication for comparison since it is fair to all algorithms.

3.2 Evaluation on Term Extraction

For comparison, three reference algorithms are used in the evaluation. The first algorithm is *TV_LinkA* which takes *CS* and *CD* into consideration and performs well (Yang et al., 2008). The second one is a supervised learning approach based on a *SVM* classifier, SVM^{light} (Joachims, 1999). Internal and external features are used by SVM^{light}. The third algorithm is the popular used *TF-IDF* algorithm. All the reference algorithms require no training except SVM^{light}. Two training sets containing thousands of positive and negative examples from IT domain and legal domain are constructed for the *SVM* classifier. The training and testing sets are not overlapped.

Table 1 and Table 2 show the performance of the proposed algorithms using different features for IT domain and legal domain, respectively. The algorithm using *CD* alone is the same as the *TF-IDF* algorithm. The algorithm using *CS* and *CD* is the *TV_LinkA* algorithm.

Algorithms	Precision (%)	Recall (%)	F-value (%)
SVM	63.6	49.5	55.6
CC	47.1	36.5	41.2
CS	65.6	51	57.4
CD(TF-IDF)	64.8	50.4	56.7
CC+CS	80.4	62.5	70.3
CC+CD	49	38.1	42.9
CS+CD (TV_LinkA)	75.4	58.6	66
CC+CS+CD	82.8	64.4	72.4

Table 1. Performance on IT Domain

Algorithms	Precision (%)	Recall (%)	F-value (%)
SVM	60.1	54.2	57.3
CC	45.2	40.3	42.6
CS	70.5	40.1	51.1
CD(TF-IDF)	59.4	52.9	56
CC+CS	64.2	49.9	56.1
CC+CD	48.4	43.1	45.6
CS+CD (TV_LinkA)	67.4	60.1	63.5
CC+CS+CD	70.2	62.6	66.2

Table 2. Performance on Legal Domain

Table 1 and Table 2 show that the proposed algorithms achieve similar performance on both domains. The proposed algorithm using all three features (*CC+CS+CD*) performs the best. The results confirm that the proposed approach are quite stable across domains and the relevance between candidates are efficient for improving performance of term extraction in different domains. The algorithm using *CC* only does not achieve good performance. Neither does *CC+CS*. The main reason is that the term candidates used in the experiments are extracted using the *TCE_DI* algorithm which can extract candidates with low statistical significance. *TCE_DI* provides a better compromise between recall and precision. *CC* alone is vulnerable to noisy candidates since it relies on the relevance between candidates themselves. However, as an additional feature to the combined use of *CS* and *CD* (*TV_LinkA*), improvement of over 10% on F-value is obtained for the IT domain, and 5% for the legal domain. This is because the noise data are eliminated by *CS* and *CD*, and *CC* help to identify additional terms that may not be statistically significant.

4 Conclusion and Future Work

In conclusion, this paper exploits the relevance between term candidates as an additional feature for term extraction approach. The proposed approach requires no prior domain knowledge and no adaptation for new domains. Experiments for term extraction are conducted on IT domain and legal domain, respectively. Evaluations indicate that the proposed algorithm using different types of relevance achieves the best performance in both domains without training.

In this work, only co-occurrence in a sentence is used as the relevance between term candidates. Other features such as syntactic relations can also be exploited. The performance may be further improved by using more efficient combination strategies. It would also be interesting to apply this approach to other languages such as English.

Acknowledgement: The project is partially supported by the Hong Kong Polytechnic University (PolyU CRG G-U297)

References

Chang Jing-Shin. 2005. Domain Specific Word Extraction from Hierarchical Web Documents: A First Step toward Building Lexicon Trees from Web Corpora. In *Proc of the 4th SIGHAN Workshop on Chinese Language Learning*: 64-71.

Eibe Frank, Gordon. W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-specific Keyphrase Extraction. In *Proc.of 16th Int. Joint Conf. on AI, IJCAI-99*: 668-673.

Joachims T. 2000. Estimating the Generalization Performance of a SVM Efficiently. In *Proc. of the Int Conf. on Machine Learning*, Morgan Kaufman, 2000.

Kageura K., and B. Umino. 1996. Methods of automatic term recognition: a review. *Term* 3(2):259-289.

Kleinberg J. 1997. Authoritative sources in a hyperlinked environment. In *Proc. of the 9th ACM-SIAM Symposium on Discrete Algorithms*: 668-677. New Orleans, America, January 1997.

Ji Luning, and Qin Lu. 2007. Chinese Term Extraction Using Window-Based Contextual Information. In *Proc. of CICLing 2007, LNCS 4394*: 62 – 74.

Li Hongqiao, Chang-Ning Huang, Jianfeng Gao, and Xiaozhong Fan. The Use of SVM for Chinese New Word Identification. In *Proc. of the 1st Int.Joint Conf. on NLP (IJCNLP2004)*: 723-732. Hainan Island, China, March 2004.

Salton, G., and McGill, M.J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

S. Brin, L. Page. The anatomy of a large-scale hypertextual web search engine. The 7th Int. World Wide Web Conf, Brisbane, Australia, April 1998, 107-117.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer, 1995.

Yang Yuhang, Qin Lu, Tiejun Zhao. (2008). Chinese Term Extraction Using Minimal Resources. The 22nd Int. Conf. on Computational Linguistics (Coling 2008). Manchester, Aug., 2008, 1033-1040.

Zhou GD, Shen D, Zhang J, Su J, and Tan SH. 2005. Recognition of Protein/Gene Names from Text using an Ensemble of Classifiers. *BMC Bioinformatics* 2005, 6(Suppl 1)**:**S7.

iChi: a bilingual dictionary generating tool

Varga István
Yamagata University,
Graduate School of Science and Engineering
dyn36150@dip.yz.yamagata-u.ac.jp

Yokoyama Shoichi
Yamagata University,
Graduate School of Science and Engineering
yokoyama@yz.yamagata-u.ac.jp

Abstract

In this paper we introduce a bilingual dictionary generating tool that does not use any large bilingual corpora. With this tool we implement our novel pivot based bilingual dictionary generation method that uses mainly the WordNet of the pivot language to build a new bilingual dictionary. We propose the usage of WordNet for good accuracy, introducing also a double directional selection method with local thresholds to maximize recall.

1 Introduction

Bilingual dictionaries are an essential, perhaps even indispensable tool not only as resources for machine translation, but also in every day activities or language education. While such dictionaries are available to and from numerous widely used languages, less represented language pairs have rarely a reliable dictionary with good coverage. The need for bilingual dictionaries for these less common language pairs is increasing, but qualified human resources are scarce. Considering that in these conditions manual compilation is highly costly, alternative methods are imperative.

Pivot language based bilingual dictionary generation is one plausible such alternative (Tanaka and Umemura, 1994; Sjöbergh, 2005; Shirai and Yamamoto, 2001; Bond and Ogura, 2007). These methods do not use large bilingual corpora, thus being suitable for low-resourced languages.

Our paper presents iChi, the implementation of our own method, an easy-to-use, customizable tool that generates a bilingual dictionary.

The paper is structured as follows: first we briefly describe the methodological background of our tool, after which we describe its basic functions, concluding with discussions. Thorough description and evaluation, including comparative analysis, are available in Varga and Yokoyama (2009).

2 Methodological background

2.1 Pivot based dictionary generation

Pivot language based bilingual dictionary generation methods rely on the idea that the lookup of a word in an uncommon language through a third, intermediated language can be automated. Bilingual dictionaries to a third, intermediate language are used to link the source and target words. The pivot language translations of the source and target head words are compared, the suitability of the source-target word pair being estimated based on the extent of the common elements.

There are two known problems of conventional pivot methods. First, a global threshold is used to determine correct translation pairs. However, the scores highly depend on the entry itself or the number of translations in the intermediate language, therefore there is a variance in what that score represents. Second, current methods perform a strictly lexical overlap of the source-intermediate and target-intermediate entries. Even if the translations from the source and target languages are semantically transferred to the intermediate language, lexically it is rarely the case. However, due to the different word-usage or paraphrases, even semantically identical or very similar words can have different definitions in different dictionaries. As a result, because of the lexical characteristic of their overlap, current methods cannot identify the differences between totally different definitions resulted by unrelated concepts, and differences in only nuances resulted by lexicographers describing the same concept, but with different words.

2.2 Specifics of our method

To overcome the limitations, namely low precision of previous pivot methods, we expand the translations in the intermediate language using

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 217–220,
Suntec, Singapore, 4 August 2009.

information extracted from *WordNet* (Miller et. al., 1990). We use the following information: *sense description, synonymy*, *antonymy* and *semantic categories*, provided by the tree structure of nouns and verbs.

To improve recall, we introduce *bidirectional selection.* As we stated above, the global threshold eliminates a large number of good translation pairs, resulting in a low recall. As a solution, we can group the translations that share the same source or target entry, and set *local thresholds* for each head word. For example, for a source language head word *entry_source* there could be multiple target language candidates: *entry_target*$_1$*, … ,entry_target*$_n$. If the top scoring *entry_target*$_k$ candidates are selected, we ensure that at least one translation will be available for *entry_source*, maintaining a high recall. Since we can group the entries in the source language and target language as well, we perform this selection twice, once in each direction. Local thresholds depend on the top scoring *entry_target*, being set to *maxscore*·*c*. Constant *c* varies between 0 and 1, allowing a small window for not maximum, but high scoring candidates. It is language and selection method dependent (See 3.2 for details).

2.3 Brief method description

First, using the source-pivot and pivot-target dictionaries, we connect the source (*s*) and target (*t*) entries that share at least one common translation in the intermediate (*i*) language. We consider each such source-target pair a *translation candidate*. Next we eliminate erroneous candidates. We examine the translation candidates one by one, looking up the source-pivot and target-pivot dictionaries, comparing pivot language translations. There are six types of translations that we label *A*-*F* and explain below as follows.

First, we select translation candidates whose translations into the intermediate language match perfectly (*type A* translations).

For most words WordNet offers *sense description* in form of synonyms for most of its senses. For a given translation candidate (*s,t*) we look up the source-pivot and target-pivot translations ($s \rightarrow I = \{s \rightarrow i_1, \ldots, s \rightarrow i_n\}$, $t \rightarrow I = \{t \rightarrow i_1, \ldots, t \rightarrow i_m\}$). We select the elements that are common in the two definitions ($I' = (s \rightarrow I) \cap (t \rightarrow I)$) and we attempt to identify their respective senses from WordNet ($sns(I')$), comparing each synonym in the WordNet's synonym description with each word from the pivot translations. As a result, we arrive at a certain set of senses from the source-pivot definitions ($sns((s \rightarrow I'))$) and target-pivot definitions ($sns((t \rightarrow I'))$). We mark $score_B(s,t)$ the Jaccard coefficient of these two sets. Scores that pass a global threshold (0.1) are selected as translation pairs. Since synonymy information is available for nouns (N), verbs (V), adjectives (A) and adverbs (R), four separate scores are calculated for each POS *(type B).*

$$score_B(s,t) = \max_{i' \in s \rightarrow I \cap t \rightarrow I} \frac{|sns(s \rightarrow i') \cap sns(t \rightarrow i')|}{|sns(s \rightarrow i') \cup sns(t \rightarrow i')|} \quad (1)$$

We expand the source-to-pivot and target-to-pivot definitions with information from WordNet (synonymy, antonymy and semantic category). The similarity of the two expanded pivot language descriptions gives a better indication on the suitability of the translation candidate. Since the same word or concept's translations into the pivot language also share the same semantic value, the extension with *synonyms* ($ext(l \rightarrow i) = (l \rightarrow i) \cup syn(l \rightarrow i)$, where $l = \{s,t\}$) the extended translation should share more common elements *(type C).*

In case of antonymy, we expand the initial definitions with the *antonyms of the antonyms* ($ext(l \rightarrow i) = (l \rightarrow i) \cup ant(ant(l \rightarrow i))$, where $l = \{s,t\}$). This extension is different from the synonymy extension, in most cases the resulting set of words being considerably larger *(type D).*

Synonymy and antonymy information are available for nouns, verbs, adjectives and adverbs, thus four separate scores are calculated for each POS.

Semantic categories are provided by the tree structure (hypernymy/hyponymy) of nouns and verbs of WordNet. We transpose each entry from the pivot translations to its semantic category ($ext(l \rightarrow i) = (l \rightarrow i) \cup semcat(l \rightarrow i)$, where $l = \{s,t\}$). We assume that the correct translation pairs share a high percentage of semantic categories.

Local thresholds are set based on the best scoring candidate for a given entry. The thresholds were *maxscore*·0.9 for synonymy and antonymy; and *maxscore*·0.8 for the semantic categories (see §3.2 for details).

$$score_{C,D,E}(s,t) = \frac{|ext(s \rightarrow i) \cap ext(t \rightarrow i)|}{|ext(s \rightarrow i) \cup ext(t \rightarrow i)|} \quad (2)$$

For a given entry, the three separate candidate lists of type C, D and E selection methods resulted in slightly different results. The good translations were among the top scoring ones, but not always scoring best. To correct this fault, a *combined selection* method is performed combining these lists. For every translation candidate we select the maximum score ($score_{rel}(s,t)$) from

the several POS (noun, verb, adjective and adverb for synonymy and antonymy relations; noun and verb for semantic category) based scores, multiplied by a multiplication factor (*mfactor*). This factor varies between 0 and 1, awarding the candidates that were selected both times during the double directional selection; and punishing when selection was made only in a single direction. c_1, c_2 and c_3 are adjustable language dependent constants, the defaults being 1, 0.5 and 0.8, respectively *(type F)*.

$$score_F(s,t) = \prod_{rel} \left(\begin{matrix} (c_1 + \max(score_{rel}(s,t))) \cdot \\ (c_2 + c_3 \cdot mfactor_{rel}(s,t)) \end{matrix} \right) \qquad (3)$$

2.4 Evaluation

We generated a Japanese-Hungarian dictionary using selection methods A, B and F; with C, D and E contributing indirectly through F.

(a) Recall evaluation

We used a Japanese frequency dictionary that we generated from the Japanese EDR corpus (Isahara, 2007) to weight each Japanese entry. Setting the standard to the frequency dictionary (its recall value being 100), we automatically search each entry from the frequency dictionary, verifying whether or not it is included in the bilingual dictionary. If it is recalled, we weight it with its frequency from the frequency dictionary.

Our method maintains the recall value of the initial translation candidates, owing to the bidirectional selection method with local thresholds. However, the recall value of a manually created Japanese-English dictionary is higher than any automatically generated dictionary's value (Table 1).

method	recall
our method	51.68
initial candidates	51.68
Japanese-English(*)	73.23

Table 1: Recall evaluation results (* marks a manually created dictionary)

(b) 1-to-1 precision evaluation

We evaluated 2000 randomly selected translation pairs, manually scoring them as *correct* (the translation conveys the same meaning, or the meanings are slightly different, but in a certain context the translation is possible: 79.15%), *undecided* (the translation pair's semantic value is similar, but a translation based on them would be faulty: 6.15%) or *wrong* (the translation pair's two entries convey a different meaning: 14.70%).

(c) 1-to-multiple evaluation

With 1-to-multiple evaluation we quantify the true reliability of the dictionary: when looking up the meanings or translations of a certain keyword, the user, whether he's a human or a machine, expects all translations to be accurate. We evaluated 2000 randomly selected Japanese entries from the initial translation candidates, scoring all Hungarian translations as *correct* (all translations are correct: 71.45%), *acceptable* (the good translations are predominant, but there are up to 2 erroneous translations: 13.85%), *wrong* (the number or wrong translations exceeds 2: 14.70%).

3 iChi

iChi is an implementation of our method. Programmed in Java, it is a platform-independent tool with a user friendly graphical interface (Image 1). Besides the MySql database it consists of: iChi.jar (java executable), iChi.cfg (configuration file), iChi.log (log file) and iChip.jar (parameter estimation tool). The major functions of iChi are briefly explained below.

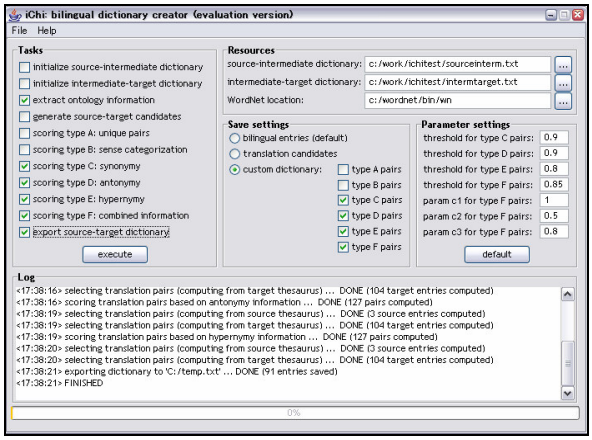


Image 1: User interface of iChi

3.1 Resources

The two bilingual dictionaries used as resources are text files, with a translation pair in each line:

```
source entry 1@pivot entry 1
source entry 2@pivot entry 2
```

The location of the pivot language's WordNet also needs to be specified. All paths are stored in the configuration file.

3.2 Parameter settings

iChip.jar estimates language dependent parameters needed for the selection methods. Its single argument is a text file that contains marked (correct: $+ or incorrect: $-) translation pairs:

```
$+source entry 1@correct target entry 1
$-source entry 2@incorrect target entry 2
```

The parameter estimation tool experiments with various threshold settings on the same (correct or incorrect) source entries. For example, with Hungarian-Japanese we considered all translation candidates whose Hungarian entry starts with "zs" (IPA: ʒ). 133 head words totalling 515 translation candidates comprise this set, 273 entries being marked as *correct*. iChip experimented with a number of thresholds to determine which ones provide with the best F-scores, e.g. retain most marked correct translations (Table 2). The F-scores were determined as follows: for example using synonymy information (type C) in case of threshold=0.85%, 343 of the 515 translation pairs were above the threshold. Among these, 221 were marked as correct, thus the precision being 221/343·100=64.43 and the recall being 221/273·100=80.95. F-score is the harmonic mean of precision and recall (71.75 in this case).

selection type	**threshold value** (%)				
	0.75	**0.80**	**0.85**	**0.90**	**0.95**
C	70.27	70.86	71.75	**72.81**	66.95
D	69.92	70.30	70.32	**70.69**	66.66
E	73.71	**74.90**	72.52	71.62	65.09
F	78.78	79.07	**79.34**	78.50	76.94

Table 2: Selection type F-scores with varying thresholds (best scores in bold)

The output is saved into the configuration file. If no parameter estimation data is available, the parameters estimated using Hungarian-Japanese are used as default.

3.3 Save settings

The generated source-target dictionary is saved into a text file that uses the same format described in §3.1. The output can be customized by choosing the desired selection methods. The default value is a dictionary with selection types A, B and F; selection types C, D and E are used only indirectly with type F.

3.4 Tasks

The tasks are run sequentially, every step being saved in the internal database, along with being logged into the log file.

4 Discussion

If heavily unbalanced resources dictionaries are used, due to the bidirectional selection method many erroneous entries will be generated. If one polysemous pivot entry has multiple translations into the source, but only some of them are translated into the target languages, unique, but incorrect source-target pairs will be generated. For example, with an English pivoted dictionary that has multiple translation of *'bank'* onto the source (*'financial institution', 'river bank'*), but only one into the target language *('river bank')*, the incorrect source*('financial institution')*-target*('river bank')* pair will be generated, since target*('river bank')* has no other alternative.

Thorough discussion on recall and precision problems concerning the methodology of iChi, are available in Varga and Yokoyama (2009).

5 Conclusions

In this paper we presented iChi, a user friendly tool that uses two dictionaries into a third, intermediate language together with the WordNet of that third language to generate a new dictionary. We briefly described the methodology, together with the basic functions. The tool is freely available online (http://mj-nlp.homeip.net/ichi).

References

Bond, F., Ogura, K. 2007. Combining linguistic resources to create a machine-tractable Japanese-Malay dictionary, *Language Resources and Evaluation*, 42(2), pp. 127-136.

Breen, J.W. 1995. Building an Electric Japanese-English Dictionary, *Japanese Studies Association of Australia Conference*, Brisbane, Queensland, Australia.

Isahara, H. (2007). EDR Electronic Dictionary – present status (EDR 電子化辞書の現状), NICT-EDR symposium, pp. 1-14. (in Japanese)

Miller G.A., Beckwith R., Fellbaum C., Gross D., Miller K.J. (1990). Introduction to WordNet: An Online Lexical Database, *Int J Lexicography* 3(4), pp. 235-244.

Sjöbergh, J. 2005. Creating a free Japanese-English lexicon, *Proceedings of PACLING*, pp. 296-300.

Shirai, S., Yamamoto, K. 2001. Linking English words in two bilingual dictionaries to generate another pair dictionary, *ICCPOL-2001*, pp. 174-179.

Tanaka, K., Umemura, K. 1994. Construction of a bilingual dictionary intermediated by a third language, *Proceedings of COLING-94*, pp. 297-303.

Varga, I., Yokoyama, S. 2009. Bilingual dictionary generation for low-resourced language pairs, *Proceedings of EMNLP 2009.*

CATiB: The Columbia Arabic Treebank

Nizar Habash and Ryan M. Roth
Center for Computational Learning Systems
Columbia University, New York, USA
{habash,ryanr}@ccls.columbia.edu

Abstract

The Columbia Arabic Treebank (CATiB) is a database of syntactic analyses of Arabic sentences. CATiB contrasts with previous approaches to Arabic treebanking in its emphasis on speed with some constraints on linguistic richness. Two basic ideas inspire the CATiB approach: no annotation of redundant information and using representations and terminology inspired by traditional Arabic syntax. We describe CATiB's representation and annotation procedure, and report on inter-annotator agreement and speed.

1 Introduction and Motivation

Treebanks are collections of manually-annotated syntactic analyses of sentences. They are primarily intended for building models for statistical parsing; however, they are often enriched for general natural language processing purposes. For Arabic, two important treebanking efforts exist: the Penn Arabic Treebank (PATB) (Maamouri et al., 2004) and the Prague Arabic Dependency Treebank (PADT) (Smrž and Hajič, 2006). In addition to syntactic annotations, both resources are annotated with rich morphological and semantic information such as full part-of-speech (POS) tags, lemmas, semantic roles, and diacritizations. This allows these treebanks to be used for training a variety of applications other than parsing, such as tokenization, diacritization, POS tagging, morphological disambiguation, base phrase chunking, and semantic role labeling.

In this paper, we describe a new Arabic treebanking effort: the Columbia Arabic Treebank (CATiB).[1] CATiB is motivated by the following three observations. First, as far as parsing Arabic research, much of the non-syntactic rich annotations are not used. For example, PATB has over 400 tags, but they are typically reduced to around 36 tags in training and testing parsers (Kulick et al., 2006). The reduction addresses the fact that sub-tags indicating case and other similar features are essentially determined syntactically and are hard to automatically tag accurately. Second, under time restrictions, the creation of a treebank faces a tradeoff between linguistic richness and treebank size. The richer the annotations, the slower the annotation process, the smaller the resulting treebank. Obviously, bigger treebanks are desirable for building better parsers. Third, both PATB and PADT use complex syntactic representations that come from modern linguistic traditions that differ from Arabic's long history of syntactic studies. The use of these representations puts higher requirements on the kind of annotators to hire and the length of their initial training.

CATiB contrasts with PATB and PADT in putting an emphasis on annotation speed for the specific task of parser training. Two basic ideas inspire the CATiB approach. First, CATiB avoids annotation of redundant linguistic information or information not targeted in current parsing research. For example, nominal case markers in Arabic have been shown to be automatically determinable from syntax and word morphology and needn't be manually annotated (Habash et al., 2007a). Also, phrasal co-indexation, empty pronouns, and full lemma disambiguation are not currently used in parsing research so we do not include them in CATiB. Second, CATiB uses a simple intuitive dependency representation and terminology inspired by Arabic's long tradition of syntactic studies. For example, CATiB relation labels include *tamyiz* (specification) and *idafa* (possessive construction) in addition to universal predicate-argument structure labels such as *subject*, *object* and *modifier*. These representation choices make it easier to train annotators without being restricted to hire people who have degrees in linguistics.

This paper briefly describes CATiB's representation and annotation procedure, and reports on produced data, achieved inter-annotator agreement and annotation speeds.

[1]This work was supported by Defense Advanced Research Projects Agency Contract No. HR0011-08-C-0110.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 221–224,
Suntec, Singapore, 4 August 2009.

2 CATiB: Columbia Arabic Treebank

CATiB uses the same basic tokenization scheme used by PATB and PADT. However, the CATiB POS tag set is much smaller than the PATB's. Whereas PATB uses over 400 tags specifying every aspect of Arabic word morphology such as definiteness, gender, number, person, mood, voice and case, CATiB uses 6 POS tags: **NOM** (non-proper nominals including nouns, pronouns, adjectives and adverbs), **PROP** (proper nouns), **VRB** (active-voice verbs), **VRB-PASS** (passive-voice verbs), **PRT** (particles such as prepositions or conjunctions) and **PNX** (punctuation).[2]

CATiB's dependency links are labeled with one of eight relation labels: **SBJ** (subject of verb or topic of simple nominal sentence), **OBJ** (object of verb, preposition, or deverbal noun), **TPC** (topic in complex nominal sentences containing an explicit pronominal referent), **PRD** (predicate marking the complement of the extended copular constructions for *kAn*[3] كان واخواتها and *An* ان واخواتها), **IDF** (relation between the possessor [dependent] to the possessed [head] in the idafa/possesive nominal construction), **TMZ** (relation of the specifier [dependent] to the specified [head] in the tamyiz/specification nominal constructions), **MOD** (general modifier of verbs or nouns), and **—** (marking *flatness* inside constructions such as first-last proper name sequences). This relation label set is much smaller than the twenty or so dashtags used in PATB to mark syntactic and semantic functions. No empty categories and no phrase co-indexation are made explicit. No semantic relations (such as time and place) are annotated.

Figure 1 presents an example of a tree in CATiB annotation. In this example, the verb زاروا *zArwA* 'visited' heads a subject, an object and a prepositional phrase. The subject includes a complex number construction formed using idafa and tamyiz and headed by the number خمسون *xmswn* 'fifty', which is the only carrier of the subject's syntactic nominative case here. The preposition في *fy* heads the prepositional phrase, whose object is a proper noun, تموز *tmwz* 'July' with an adjectival modifier, الماضي *AlmADy* 'last'. See Habash et al. (2009) for a full description of CATiB's guidelines and a detailed comparison with PATB and PADT.

[2]We are able to reproduce a parsing-tailored tag set [size 36] (Kulick et al., 2006) automatically at 98.5% accuracy using features from the annotated trees. Details of this result will be presented in a future publication.

[3]Arabic transliterations are in the Habash-Soudi-Buckwalter transliteration scheme (Habash et al., 2007b).

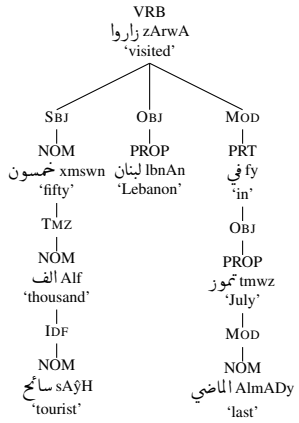


Figure 1: CATiB annotation for the sentence خمسون الف سائح زاروا لبنان في تموز الماضي
xmswn Alf sAŷH zArwA lbnAn fy tmwz AlmADy
'50 thousand tourists visited Lebanon last July.'

3 Annotation Procedure

Although CATiB is independent of previous annotation projects, it builds on existing resources and lessons learned. For instance, CATiB's pipeline uses PATB-trained tools for tokenization, POS-tagging and parsing. We also use the TrEd annotation interface developed in coordination with the PADT. Similarly, our annotation manual is guided by the wonderfully detailed manual of the PATB for coverage (Maamouri et al., 2008).

Annotators Our five annotators and their supervisor are all educated native Arabic speakers. Annotators are hired on a part-time basis and are not required to be on-site. The annotation files are exchanged electronically. This arrangement allows more annotators to participate, and reduces logistical problems. However, having no full-time annotators limits the overall weekly annotation rate. Annotator training took about two months (150 hrs/annotator on average). This training time is much shorter than the PATB's six-month training period.[4]

Below, we describe our pipeline in some detail including the different resources we use.

Data Preparation The data to annotate is split into *batches* of 3-5 documents each, with each document containing around 15-20 sentences (400-600 tokens). Each annotator works on one batch at a time. This procedure and the size of the batches was determined to be optimal for both the software and the annotators' productivity. To track the annotation quality, several key documents are selected for inter-annotator agreement (IAA) checks. The IAA documents are chosen to

[4]Personal communication with Mohamed Maamouri.

cover a range of sources and to be of average document size. These documents (collectively about 10% of the token volume) are seeded throughout the batches. Every annotator eventually annotates each one of the IAA documents, but is never told which documents are for IAA.

Automatic Tokenization and POS Tagging We use the MADA&TOKAN toolkit (Habash and Rambow, 2005) for initial tokenization and POS tagging. The tokenization F-score is 99.1% and the POS tagging accuracy (on the CATiB POS tag set; with gold tokenization) is above 97.7%.

Manual Tokenization Correction Tokenization decisions are manually checked and corrected by the annotation supervisor. New POS tags are assigned manually only for corrected tokens. Full POS tag correction is done as part of the manual annotation step (see below). The speed of this step is well over 6K tokens/hour.

Automatic Parsing Initial dependency parsing in CATiB is conducted using MaltParser (Nivre et al., 2007). An initial parsing model was built using an automatic constituency-to-dependency conversion of a section of PATB *part 3* (PATB3-Train, 339K tokens). The quality of the automatic conversion step is measured against a hand-annotated version of an automatically converted held-out section of PATB3 (PATB3-Dev, 31K tokens). The results are 87.2%, 93.16% and 83.2% for attachment (**ATT**), label (**LAB**) and labeled attachment (**LABATT**) accuracies, respectively. These numbers are 95%, 98% and 94% (respectively) of the IAA scores on that set.[5] At the production midpoint another parsing model was trained by adding all the CATiB annotations generated up to that point (513K tokens total). An evaluation of the parser against the CATiB version of PATB3-Dev shows the **ATT**, **LAB** and **LABATT** accuracies are 81.7%, 91.1% and 77.4% respectively.[6]

Manual Annotation CATiB uses the TrEd tool as a visual interface for annotation.[7] The parsed trees are converted to TrEd format and delivered to the annotators. The annotators are asked to only correct the POS, syntactic structure and relation labels. Once annotated (i.e. corrected), the documents are returned to be packaged for release.

[5]Conversion will be discussed in a future publication.

[6]Since CATiB POS tag set is rather small, we extend it automatically deterministically to a larger tag set for parsing purposes. Details will be presented in a future publication.

[7]http://ufal.mff.cuni.cz/∼pajas/tred

IAA Set	Sents	POS	ATT	LAB	LABATT
PATB3-Dev	All	98.6	91.5	95.3	88.8
	≤ 40	98.7	91.7	94.7	88.6
PROD	All	97.6	89.2	93.0	85.0
	≤ 40	97.7	91.5	94.1	87.7

Table 1: Average pairwise IAA accuracies for 5 annotators. The **Sents** column indicates which sentences were evaluated, based on token length. The sizes of the sets are 2.4K (PATB3-Dev) and 3.8K (PROD) tokens.

4 Results

Data Sets CATiB annotated data is taken from the following LDC-provided resources:[8] LDC2007E46, LDC2007E87, GALE-DEV07, MT05 test set, MT06 test set, and PATB (part 3). These datasets are 2004-2007 newswire feeds collected from different news agencies and news papers, such as Agence France Presse, Xinhua, Al-Hayat, Al-Asharq Al-Awsat, Al-Quds Al-Arabi, An-Nahar, Al-Ahram and As-Sabah. The CATiB-annotated PATB3 portion is extracted from An-Nahar news articles from 2002. Headlines, datelines and bylines are not annotated and some sentences are excluded for excessive (>300 tokens) length and formatting problems. Over 273K tokens (228K words, 7,121 trees) of data were annotated, not counting IAA duplications. In addition, the PATB part 1, part 2 and part 3 data is automatically converted into CATiB representation. This converted data contributes an additional 735K tokens (613K words, 24,198 trees). Collectively, the CATiB version 1.0 release contains over 1M tokens (841K words, 31,319 trees), including annotated and converted data.

Annotator Speeds Our POS and syntax annotation rate is 540 tokens/hour (with some reaching rates as high as 715 tokens/hour). However, due to the current part-time arrangement, annotators worked an average of only 6 hours/week, which meant that data was annotated at an average rate of 15K tokens/week. These speeds are much higher than reported speeds for complete (POS+syntax) annotation in PATB (around 250-300 tokens/hour) and PADT (around 75 tokens/hour).[9]

Basic Inter-Annotator Agreement We present IAA scores for **ATT**, **LAB** and **LABATT** on IAA

[8]http://www.ldc.upenn.edu/

[9]Extrapolated from personal communications, Mohamed Maamouri and Otakar Smrž. In the PATB, the syntactic annotation step alone has similar speed to CATiB's full POS and syntax annotation. The POS annotation step is what slows down the whole process in PATB.

IAA File	Toks/hr	POS	ATT	LAB	LABATT
HI	398	97.0	94.7	96.1	91.2
HI-S	956	97.0	97.8	97.9	95.7
LO	476	98.3	88.8	91.7	82.3
LO-S	944	97.7	91.0	93.8	85.8

Table 2: Highest and lowest average pairwise IAA accuracies for 5 annotators achieved on a single document – before and after serial annotation. The "-S" suffix indicates the result after the second annotation.

subsets from two data sets in Table 1: PATB3-Dev is based on an automatically converted PATB set and PROD refers to all the new CATiB data. We compare the IAA scores for all sentences and for sentences of token length ≤ 40 tokens. The IAA scores in PROD are lower than PATB3-Dev, this is understandable given that the error rate of the conversion from a manual annotation (starting point of PATB3-Dev) is lower than parsing (starting point for PROD). Length seems to make a big difference in performance for PROD, but less so for PATB3-Dev, which makes sense given their origins. Annotation training did not include very long sentences. Excluding long sentences during production was not possible because the data has a high proportion of very long sentences: for PROD set, 41% of sentences had >40 tokens and they constituted over 61% of all tokens.

The best reported IAA number for PATB is 94.3% F-measure after extensive efforts (Maamouri et al., 2008). This number does not include dashtags, empty categories or indices. Our numbers cannot be directly compared to their number because of the different metrics used for different representations.

Serial Inter-Annotator Agreement We test the value of *serial annotation*, a procedure in which the output of annotation is passed again as input to another annotator in an attempt to improve it. The IAA documents with the highest (HI, 333 tokens) and lowest (LO, 350 tokens) agreement scores in PROD are selected. The results, shown in Table 2, indicate that serial annotation is very helpful reducing **LABATT** error by 20-50%. The reduction in LO is not as large as that in HI, unfortunately. The second round of annotation is almost twice as fast as the first round. The overall reduction in speed (end-to-end) is around 30%.

Disagreement Analysis We conduct an error analysis of the basic-annotation disagreements in HI and LO. The two sets differ in sentence length, source and genre: HI has 28 tokens/sentence and contains AFP general news, while LO has 58 tokens/sentence and contains Xinhua financial news. The most common POS disagreement in both sets is NOM/PROP confusion, a common issue in Arabic POS tagging in general. The most common attachment disagreements in LO are as follows: prepositional phrase (PP) and nominal modifiers (8% of the words had at least one dissenting annotation), complex constructions (dates, proper nouns, numbers and currencies) (6%), subordination/coordination (4%), among others. The respective proportions for HI are 5%, 5% and 1%. Label disagreements are mostly in nominal modification (MOD/TMZ/IDF/—) (LO 10%, HI 5% of the words had at least one dissenting annotation).

The error differences between HI and LO seem to primarily correlate with length difference and less with genre and source differences.

5 Conclusion and Future Work

We presented CATiB, a treebank for Arabic parsing built with faster annotation speed in mind. In the future, we plan to extend our annotation guidelines focusing on longer sentences and specific complex constructions, introduce serial annotation as a standard part of the annotation pipeline, and enrich the treebank with automatically generated morphological information.

References

N. Habash, R. Faraj and R. Roth. 2009. Syntactic Annotation in the Columbia Arabic Treebank. In *Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

N. Habash and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *ACL'05*, Ann Arbor, Michigan.

N. Habash, R. Gabbard, O. Rambow, S. Kulick, and M. Marcus. 2007a. Determining case in Arabic: Learning complex linguistic behavior requires complex linguistic features. In *EMNLP'07*, Prague, Czech Republic.

N. Habash, A. Soudi, and T. Buckwalter. 2007b. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology*. Springer.

S. Kulick, R. Gabbard, and M. Marcus. 2006. Parsing the Arabic Treebank: Analysis and Improvements. In *Treebanks and Linguistic Theories Conference*, Prague, Czech Republic.

M. Maamouri, A. Bies, and T. Buckwalter. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

M. Maamouri, A. Bies and S. Kulick. 2008. Enhancing the Arabic treebank: a collaborative effort toward new annotation guidelines. In *LREC'08*, Marrakech, Morocco.

J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kubler, S. Marinov, and E. Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

O. Smrž and J. Hajič. 2006. The Other Arabic Treebank: Prague Dependencies and Functions. In Ali Farghaly, editor, *Arabic Computational Linguistics*. CSLI Publications.

A Beam-Search Extraction Algorithm for Comparable Data

Christoph Tillmann
IBM T.J. Watson Research Center
Yorktown Heights, N.Y. 10598
ctill@us.ibm.com

Abstract

This paper extends previous work on extracting parallel sentence pairs from comparable data (Munteanu and Marcu, 2005). For a given source sentence S, a maximum entropy (ME) classifier is applied to a large set of candidate target translations . A beam-search algorithm is used to abandon target sentences as non-parallel early on during classification if they fall outside the beam. This way, our novel algorithm avoids any document-level prefiltering step. The algorithm increases the number of extracted parallel sentence pairs significantly, which leads to a BLEU improvement of about 1 % on our Spanish-English data.

1 Introduction

The paper presents a novel algorithm for extracting parallel sentence pairs from comparable monolingual news data. We select source-target sentence pairs (S, T) based on a ME classifier (Munteanu and Marcu, 2005). Because the set of target sentences T considered can be huge, previous work (Fung and Cheung, 2004; Resnik and Smith, 2003; Snover et al., 2008; Munteanu and Marcu, 2005) pre-selects target sentences T at the document level . We have re-implemented a particular filtering scheme based on BM25 (Quirk et al., 2007; Utiyama and Isahara, 2003; Robertson et al., 1995). In this paper, we demonstrate a different strategy . We compute the ME score incrementally at the word level and apply a beam-search algorithm to a large number of sentences. We abandon target sentences early on during classification if they fall outside the beam. For comparison purposes, we run our novel extraction algorithm with and without the document-level prefiltering step. The results in Section 4 show that the number of extracted sentence pairs is more than doubled which also leads to an increase in BLEU by about 1 % on the Spanish-English data.

The classification probability is defined as follows:

$$p(c|S,T) = \frac{exp(\ w^T \cdot f(c,S,T)\ )}{Z(S,T)}\ , \quad (1)$$

where $S = s_1^J$ is a source sentence of length J and $T = t_1^I$ is a target sentence of length I. $c \in \{0,1\}$ is a binary variable . $p(c|S,T) \in [0,1]$ is a probability where a value $p(c=1|S,T)$ close to 1.0 indicates that S and T are translations of each other. $w \in \mathbb{R}^n$ is a weight vector obtained during training. $f(c,S,T)$ is a feature vector where the features are co-indexed with respect to the alignment variable c. Finally, $Z(S,T)$ is an appropriately chosen normalization constant.

Section 2 summarizes the use of the binary classifier. Section 3 presents the beam-search algorithm. In Section 4, we show experimental results. Finally, Section 5 discusses the novel algorithm.

2 Classifier Training

The classifier in Eq. 1 is based on several real-valued feature functions f_i . Their computation is based on the so-called IBM Model-1 (Brown et al., 1993). The Model-1 is trained on some parallel data available for a language pair, i.e. the data used to train the baseline systems in Section 4. $p(s|T)$ is the Model-1 probability assigned to a source word s given the target sentence T , $p(t|S)$ is defined accordingly. $p(s|t)$ and $p(t|s)$ are word translation probabilities obtained by two parallel Model-1 training steps on the same data, but swapping the role of source and target language. To compute these values efficiently, the implementation techniques in (Tillmann and Xu, 2009) are used. **Coverage** and **fertility** features are defined based on the Model-1 Viterbi alignment: a source

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 225–228,
Suntec, Singapore, 4 August 2009.

word s is said to be **covered** if there is a target word $t \in T$ such that its probability is above a threshold ϵ: $p(s|t) > \epsilon$. We define the **fertility** of a source word s as the number of target words $t \in T$ for which $p(s|t) > \epsilon$. Target word coverage and fertility are defined accordingly. A large number of 'uncovered' source and target positions as well as a large number of high fertility words indicate non-parallelism. We use the following $N = 7$ features: 1,2) lexical Model-1 weighting: $\sum_s -log(\ p(s|T)\ )$ and $\sum_t -log(\ p(t|S)\ )$, 3,4) number of uncovered source and target positions, 5,6) sum of source and target fertilities, 7) number of covered source and target positions . These features are defined in a way that they can be computed incrementally at the word level. Some thresholding is applied, e.g. a sequence of uncovered positions has to be at least 3 positions long to generate a non-zero feature value . In the feature vector $f(c, S, T)$, each feature f_i occurs potentially twice, once for each class $c \in \{0, 1\}$. For the feature vector $f(c = 1, S, T)$, all the feature values corresponding to class $c = 0$ are set to 0, and vice versa. This particular way of defining the feature vector is needed for the search in Section 3: the contribution of the 'negative' features for $c = 0$ is only computed when Eq. 1 is evaluated for the highest scoring final hypothesis in the beam. To train the classifier, we have manually annotated a collection of 524 sentence pairs . A sentence pair is considered parallel if at least 75 % of source and target words have a corresponding translation in the other sentence, otherwise it is labeled as non-parallel. A weight vector $w \in \mathbb{R}^{2*N}$ is trained with respect to classification accuracy using the on-line maxent training algorithm in (Tillmann and Zhang, 2007).

3 Beam Search Algorithm

We process the comparable data at the sentence level: sentences are indexed based on their publication date. For each source sentence S, a matching score is computed over all the target sentences $T_m \in \Theta$ that have a publication date which differs less than 7 days from the publication date of the source sentence [1]. We are aiming at finding the $\hat{T}$ with the highest probability $p(c = 1|S, \hat{T})$, but we cannot compute that probability for all sentence pairs (S, T_m) since $|\Theta|$ can be in tens of thousands of sentences . Instead, we use a beam-search algorithm to search for the sentence pair $(S, \hat{T})$ with the highest matching score $w^T \cdot f(1, S, \hat{T})$ [2]. The 'light-weight' features defined in Section 2 are such that the matching score can be computed incrementally while processing the source and target sentence positions in some order. To that end, we maintain a stack of matching hypotheses for each source position j. Each hypothesis is assigned a partial matching score based on the source and target positions processed so far. Whenever a partial matching score is low compared to partial matching scores of other target sentence candidates, that translation pair can be discarded by carrying out a beam-search pruning step. The search is organized in a single left-to-right run over the source positions $1 \leq j \leq J$ and all active partial hypotheses match the same portion of that source sentence. There is at most a single active hypothesis for each different target sentence T_i, and search states are defined as follows:

$$[\ m\ ,\ j\ ,\ u_j\ ,\ u_i\ ;\ d\ ]\ .$$

Here, $m \in \{1, \cdots, |\Theta|\}$ is a target sentence index. j is a position in the source sentence, u_j and u_i are the number of uncovered source and target positions to the left of source position j and target position i (coverage computation is explained above), and d is the partial matching score . The target position i corresponding to the source position j is computed deterministically as follows:

$$i \ = \ \lceil I \cdot \frac{j}{J} \rceil\ , \qquad (2)$$

where the sentence lengths I and J are known for a sentence pair (S, T). Covering an additional source position leads to covering additional target positions as well, and source and target features are computed accordingly. The search is initialized by adding a single hypothesis for each target sentence $T_m \in \Theta$ to the stack for $j = 1$:

$$[\ m\ ,\ j = 1\ ,\ u_j = 0\ ,\ u_i = 0\ ;\ 0\ ]\ .$$

During the left-to-right search , state transitions of the following type occur:

$$[\ m\ ,\ j\ ,\ u_j\ ,\ u_i\ ;\ d\ ] \rightarrow$$
$$[\ m\ ,\ j+1\ ,\ u_j^{'}\ ,\ u_i^{'}\ ;\ d'\ ]\ ,$$

[1] In addition, the sentence length filter in (Munteanu and Marcu, 2005) is used: the length ratio $max(J, I)/min(J, I)$ of source and target sentence has to be smaller than 2.

[2] This is similar to standard phrase-based SMT decoding, where a set of real-valued features is used and any sentence-level normalization is ignored during decoding. We assume the effect of this approximation to be small.

where the partial score is updated as: $d' = d + w^T \cdot f(1, j, i)$. Here, $f(1, j, i)$ is a partial feature vector computed for all the additional source and target positions processed in the last extension step. The number of uncovered source and target positions u' is updated as well. The beam-search algorithm is carried out until all source positions j have been processed. We extract the highest scoring partial hypothesis from the final stack $j = J$. For that hypothesis, we compute a global feature vector $f(1, S, T)$ by adding all the local $f(1, j, i)$'s component-wise. The 'negative' feature vector $f(0, S, T)$ is computed from $f(1, S, T)$ by copying its feature values. We then use Eq. 1 to compute the probability $p(1|S, T)$ and apply a threshold of $\theta = 0.75$ to extract parallel sentence pairs. We have adjusted beam-search pruning techniques taken from regular SMT decoding (Tillmann et al., 1997; Koehn, 2004) to reduce the number of hypotheses after each extension step. Currently, only histogram pruning is employed to reduce the number of hypotheses in each stack.

The resulting beam-search algorithm is similar to a monotone decoder for SMT: rather then incrementally generating a target translation, the decoder is used to select entire target sentences out of a pre-defined list. That way, our beam search algorithm is similar to algorithms in large-scale speech recognition (Ney, 1984; Vintsyuk, 1971), where an acoustic signal is matched to a pre-assigned list of words in the recognizer vocabulary.

4 Experiments

The parallel sentence extraction algorithm presented in this paper is tested in detail on all of the large-scale Spanish-English Gigaword data (Graff, 2006; Graff, 2007) as well as on some smaller Portuguese-English news data . For the Spanish-English data , matching sentence pairs come from the same news feed. Table 1 shows the size of the comparable data, and Table 2 shows the effect of including the additional sentence pairs into the training of a phrase-based SMT system. Here, both languages use a test set with a single reference. The test data comes from Spanish and Portuguese news web pages that have been translated into English. Including about 1.35 million sentence pairs extracted from the Gigaword data, we obtain a statistically significant improvement from 42.3 to 45.7 in BLEU. The baseline system has been trained on about 1.8 million sentence pairs from Europarl and FBIS parallel data. We also present results for a Portuguese-English system: the baseline has been trained on Europarl and JRC data. Parallel sentence pairs are extracted from comparable news data published in 2006. For this data, no document-level information was available. To gauge the effect of the document-level pre-filtering step, we have re-implemented an IR technique based on BM25 (Robertson et al., 1995). This type of pre-filtering has also been used in (Quirk et al., 2007; Utiyama and Isahara, 2003). We split the Spanish data into documents. Each Spanish document is translated into a bag of English words using Model-1 lexicon probabilities trained on the baseline data. Each of these English bag-of-words is then issued as a query against all the English documents that have been published within a 7 day window of the source document. We select the 20 highest scoring English documents for each source document . These 20 documents provide a restricted set of target sentence candidates. The sentence-level beam-search algorithm without the document-level filtering step searches through close to 1 trillion sentence pairs. For the data obtained by the BM25-based filtering step, we still use the same beam-search algorithm but on a much smaller candidate set of only 25.4 billion sentence pairs. The probability selection threshold θ is determined on some development set in terms of precision and recall (based on the definitions in (Munteanu and Marcu, 2005)). The classifier obtains an F-measure classifications performance of about 85 %. The BM25 filtering step leads to a significantly more complex processing pipeline since sentences have to be indexed with respect to document boundaries and publication date. The document-level pre-filtering reduces the overall processing time by about 40 % (from 4 to 2.5 days on a 100-CPU cluster). However, the exhaustive sentence-level search improves the BLEU score by about 1 % on the Spanish-English data.

Table 1: Corpus statistics for comparable data.

	Spanish	English
Sentences	19.4 million	47.9 million
Words	601.5 million	1.36 billion
	Portuguese	English
Sentences	366.0 thousand	5.3 million
Words	11.6 million	171.1 million

Table 2: Spanish-English and Portuguese-English extraction results. Extraction threshold is $\theta = 0.75$ for both language pairs. # cands reports the size of the overall search space in terms of sentence pairs processed .

Data Source	# cands	# pairs	Bleu
Baseline	-	$1.826\ M$	42.3
+ Giga	$999.3\ B$	$1.357\ M$	45.7
+ Giga (BM25)	$25.4\ B$	$0.609\ M$	44.8
Baseline	-	$2.222\ M$	45.3
+ News Data 2006	$77.8\ B$	$56\ K$	47.2

5 Future Work and Discussion

In this paper, we have presented a novel beam-search algorithm to extract sentence pairs from comparable data . It can avoid any pre-filtering at the document level (Resnik and Smith, 2003; Snover et al., 2008; Utiyama and Isahara, 2003; Munteanu and Marcu, 2005; Fung and Cheung, 2004). The novel algorithm is successfully evaluated on news data for two language pairs. A related approach that also avoids any document-level pre-filtering has been presented in (Tillmann and Xu, 2009). The efficient implementation techniques in that paper are extended for the ME classifier and beam search algorithm in the current paper, i.e. feature function values are cached along with Model-1 probabilities.

The search-driven extraction algorithm presented in this paper might also be applicable to other NLP extraction task, e.g. named entity extraction. Rather then employing a cascade of filtering steps, a one-stage search with a specially adopted feature set and search space organization might be carried out . Such a search-driven approach makes less assumptions about the data and may increase the number of extracted entities, i.e. increase recall.

Acknowledgments

We would like to thanks the anonymous reviewers for their valuable remarks.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *CL*, 19(2):263–311.

Pascale Fung and Percy Cheung. 2004. Mining Very-Non-Parallel Corpora: Parallel Sentence and Lexicon Extraction via Bootstrapping and EM. In *Proc, of EMNLP 2004*, pages 57–63, Barcelona, Spain, July.

Dave Graff. 2006. *LDC2006T12: Spanish Gigaword Corpus First Edition.* LDC.

Dave Graff. 2007. *LDC2007T07: English Gigaword Corpus Third Edition.* LDC.

Philipp Koehn. 2004. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Proceedings of AMTA'04*, Washington DC, September-October.

Dragos S. Munteanu and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *CL*, 31(4):477–504.

H. Ney. 1984. The Use of a One-stage Dynamic Programming Algorithm for Connected Word Recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):263–271.

Chris Quirk, Raghavendra Udupa, and Arul Menezes. 2007. Generative Models of Noisy Translations with Applications to Parallel Fragment Extraction. In *Proc. of the MT Summit XI*, pages 321–327, Copenhagen,Demark, September.

Philip Resnik and Noah Smith. 2003. The Web as Parallel Corpus. *CL*, 29(3):349–380.

S E Robertson, S Walker, M M Beaulieu, and M Gatford. 1995. Okapi at TREC-4. In *Proc. of the 4th Text Retrieval Conference (TREC-4)*, pages 73–96.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2008. Language and Translation Model Adaptation using Comparable Corpora. In *Proc. of EMNLP08*, pages 856–865, Honolulu, Hawaii, October.

Christoph Tillmann and Jian-Ming Xu. 2009. A Simple Sentence-Level Extraction Algorithm for Comparable Data. In *Companion Vol. of NAACL HLT 09*, pages 93–96, Boulder, Colorado, June.

Christoph Tillmann and Tong Zhang. 2007. A Block Bigram Prediction Model for Statistical Machine Translation. *ACM-TSLP*, 4(6):1–31, July.

Christoph Tillmann, Stefan Vogel, Hermann Ney, and Alex Zubiaga. 1997. A DP-based Search Using Monotone Alignments in Statistical Translation. In *Proc. of ACL 97*, pages 289–296, Madrid,Spain, July.

Masao Utiyama and Hitoshi Isahara. 2003. Reliable Measures for Aligning Japanese-English News Articles and Sentences. In *Proc. of ACL03*, pages 72–79, Sapporo, Japan, July.

T.K. Vintsyuk. 1971. Element-Wise Recognition of Continuous Speech Consisting of Words From a Specified Vocabulary. *Cybernetics (Kibernetica)*, (2):133–143, March-April.

Optimizing Word Alignment Combination For Phrase Table Training

Yonggang Deng *and* **Bowen Zhou**
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598, USA
{ydeng,zhou}@us.ibm.com

Abstract

Combining word alignments trained in two translation directions has mostly relied on heuristics that are not directly motivated by intended applications. We propose a novel method that performs combination as an optimization process. Our algorithm explicitly maximizes the effectiveness function with greedy search for phrase table training or synchronized grammar extraction. Experimental results show that the proposed method leads to significantly better translation quality than existing methods. Analysis suggests that this simple approach is able to maintain accuracy while maximizing coverage.

1 Introduction

Word alignment is the process of identifying word-to-word links between parallel sentences. It is a fundamental and often a necessary step before linguistic knowledge acquisitions, such as training a phrase translation table in phrasal machine translation (MT) system (Koehn et al., 2003), or extracting hierarchial phrase rules or synchronized grammars in syntax-based translation framework.

Most word alignment models distinguish translation direction in deriving word alignment matrix. Given a parallel sentence, word alignments in two directions are established first, and then they are combined as knowledge source for phrase training or rule extraction. This process is also called symmetrization. It is a common practice in most state of the art MT systems. Widely used alignment models, such as IBM Model serial (Brown et al., 1993) and HMM , all assume one-to-many alignments. Since many-to-many links are commonly observed in natural language, symmetrization is able to make up for this modeling limitation. On the other hand, combining two directional alignments practically can lead to improved performance. Symmetrization can also be realized during alignment model training (Liang et al., 2006; Zens et al., 2004).

Given two sets of word alignments trained in two translation directions, two extreme combination are intersection and union. While intersection achieves high precision with low recall, union is the opposite. A right balance of these two extreme cases would offer a good coverage with reasonable accuracy. So starting from intersection, gradually adding elements in the union by heuristics is typically used. Koehn et al. (2003) grow the set of word links by appending neighboring points, while Och and Hey (2003) try to avoid both horizontal and vertical neighbors. These heuristic-based combination methods are not driven explicitly by the intended application of the resulting output. Ayan (2005) exploits many advanced machine learning techniques for general word alignment combination problem. However, human annotation is required for supervised training in those techniques.

We propose a new combination method. Like heuristics, we aim to find a balance between intersection and union. But unlike heuristics, combination is carried out as an optimization process driven by an effectiveness function. We evaluate the impact of each alignment pair w.r.t. the target application, say phrase table training, and gradually add or remove the word link that currently can maximize the predicted benefit measured by the effectiveness function. More specifically, we consider the goal of word alignment combination is for phrase table training, and we directly motivate word alignment combination as a process of maximizing the number of phrase translations that can be extracted within a sentence pair.

2 Combination As Optimization Process

Given a parallel sentence $(\mathbf{e} = e_1^I, \mathbf{f} = f_1^J)$, a word link is represented by a pair of indices (i, j),

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 229–232,
Suntec, Singapore, 4 August 2009.

which means that Foreign word f_j is aligned with English word e_i. The direction of word alignments is ignored. Since the goal of word alignment combination is for phrase table training, we first formally define a phrase translation. Provided with a set of static word alignments $\mathbf{A}$, a phrase pair $(e_{i_1}^{i_2}, f_{j_1}^{j_2})$ is considered translation of each other if and only if there exists at least one word link between them and no cross phrase boundary links exist in $\mathbf{A}$, i.e., for all $(i,j) \in \mathbf{A}, i \in [i_1, i_2]$ iff $j \in [j_1, j_2]$. Notice that by this definition, it does not matter whether boundary words of the phrase pairs should be aligned or not. Let $PP_n(\mathbf{A})$ denote the set of phrase pairs that can be extracted with $\mathbf{A}$ where up to n boundary words are allowed to be not-aligned, i.e., aligned to empty word NULL. As can be imagined, increasing n would improve recall of phrase table but likely to hurt precision. For word alignment combination, we focus on the set with high accuracy where $n = 0$.

Let $\mathbf{A}_1$, $\mathbf{A}_2$ denote two sets of word alignments to be combined for the given sentence pair. For instance, $\mathbf{A}_1$ could be word alignments from English to foreign while $\mathbf{A}_2$ the other direction. On different setup, $\mathbf{A}_1$ could be Model-4 alignments, while $\mathbf{A}_2$ is from HMM. In the first combination method we presented in Algorithm 1, we start with intersection $\mathbf{A}_I$. $\mathbf{A}_c$ is the candidate link set to be evaluated and appended to the combined set $\mathbf{A}$. Its initial value is the difference between union and intersection. We assume that there is an effectiveness function $g(\cdot)$ which quantitatively measures the 'goodness' of a alignment set for the intended application. A higher number indicates a better alignment set. We use the function g to drive the process. Each time, we identify the best word link $(\hat{i}, \hat{j})$ in the candidate set that can maximize the function g and append it to the current set $\mathbf{A}$. This process is repeated until the candidate set is empty or adding any link in the set would lead to degradation. Finally (line 15 to 21), we pickup word links in the candidate set to align those uncovered words. This is applied to maximize coverage, which is similar as the 'final' in (Koehn et al., 2003). Again, we use the function $g(\cdot)$ to rank the word links in $\mathbf{A}_c$ and sequentially append them to $\mathbf{A}$ depending on current word coverage.

The algorithm clearly is a greedy search procedure that maximizes the function g. Since we plan to take the combined word alignments for phrase translation training, a natural choice for g is the number of phrase pairs that can be extracted with the given alignment set. We choose $g(\mathbf{A}) = |PP_0(\mathbf{A})|$, where we only count phrase pairs that all boundary words are aligned. The reason of putting a tight constraint is to maintain phrase table accuracy while improving the coverage. By keeping track of the span of currently aligned words, we can have efficient implementation of the function g.

Algorithm 1 Combination of $\mathbf{A}_1$ and $\mathbf{A}_2$ as an Optimized Expanding Process

1: $\mathbf{A}_I = \mathbf{A}_1 \cap \mathbf{A}_2$, $\mathbf{A}_U = \mathbf{A}_1 \cup \mathbf{A}_2$
2: $\mathbf{A} = \mathbf{A}_I$, $\mathbf{A}_c = \mathbf{A}_U - \mathbf{A}_I$
3: $total = g(\mathbf{A})$
4: **while** $\mathbf{A}_c \neq \emptyset$ **do**
5: $\quad curMax = \max_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
6: $\quad$ **if** $curMax \geq total$ **then**
7: $\quad\quad (\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
8: $\quad\quad \mathbf{A} = \mathbf{A} \cup \{(\hat{i}, \hat{j})\}$
9: $\quad\quad \mathbf{A}_c = \mathbf{A}_c - \{(\hat{i}, \hat{j})\}$
10: $\quad\quad total = curMax$
11: $\quad$ **else** {adding any link will make it worse}
12: $\quad\quad$ break
13: $\quad$ **end if**
14: **end while**
15: **while** $\mathbf{A}_c \neq \emptyset$ **do**
16: $\quad (\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
17: $\quad$ **if** $e_{\hat{i}}$ is not aligned or $f_{\hat{j}}$ is not aligned **then**
18: $\quad\quad \mathbf{A} = \mathbf{A} \cup \{(\hat{i}, \hat{j})\}$
19: $\quad$ **end if**
20: $\quad \mathbf{A}_c = \mathbf{A}_c - \{(\hat{i}, \hat{j})\}$
21: **end while**
22: return $\mathbf{A}$

Alternatively, the optimization can go in opposite direction. We start with the union $\mathbf{A} = \mathbf{A}_U$, and gradually remove the worse word link $(\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} - \{(i,j)\})$ that could maximize the effectiveness function. Similarly, this shrinking process is repeated until either candidate set is empty or removing any link in the candidate set would reduce the value of function g.

Other choice of 'goodness' function g is possible. For instance, one could consider syntactic constraints, or weight phrase pairs differently according to their global co-occurrence. The basic idea is to implement the combination as an iterative customized optimization process that is driven by the application.

3 Experimental Results

We test the proposed new idea on Persian Farsi to English translation. The task is to translate spoken Farsi into English. We decode reference transcription so recognition is not an issue. The training

data was provided by the DARPA TransTac program. It consists of around 110K sentence pairs with 850K English words in the military force protection domain. We train IBM Model-4 using GIZA++ toolkit (Och and Ney, 2003) in two translation directions and perform different word alignment combination. The resulting alignment set is used to train a phrase translation table, where Farsi phrases are limited to up to 6 words.

The quality of resulting phrase translation table is measured by translation results. Our decoder is a phrase-based multi-stack implementation of the log-linear model similar to Pharaoh (Koehn et al., 2003). Like other log-linear model based decoders, active features in our translation engine include translation models in two directions, lexicon weights in two directions, language model, lexicalized reordering models, sentence length penalty and other heuristics. These feature weights are tuned on the dev set to achieve optimal translation performance evaluated by automatic metric. The language model is a statistical 4-gram model estimated with Modified Kneser-Ney smoothing (Chen and Goodman, 1996) using only English sentences in the parallel training data.

3.1 Phrase Table Comparison

We first study the impact of different word alignment combination methods on phrase translation table, and compare our approaches to heuristic based methods. The same English to Farsi and Farsi to English Model-4 word alignments are used, but we try different combination methods and analysis the final alignment set and the resulting phase translation table. Table 1 presents some statistics. Each row corresponds to a particular combination. The first two are intersection (I) and union (U). The next two methods are heuristic (H) in (Och and Ney, 2003) and grow-diagonal (GD) proposed in (Koehn et al., 2003). Our proposed methods are presented in the following two rows: one is optimization as an expanding process (OE), the other is optimization as an shrinking process (OS). In the last four rows, we add 'final' operation (line 15 to 21 in Algorithm 1).

For each method, we calculate the output alignment set size as a percentage of the union (the 2nd column) and resulting phrase table ($PP_n(\mathbf{A})$) size (in thousand) with different constrain on the maximum number of unaligned boundary words $n = 0, 1, 2$ (the next 3 columns). As we can see, the intersection has less than half of all word links in the pool. This implies the underlying word alignment quality leaves much room for improvements, mainly due to data sparseness. Not surprisingly, when relaxing unaligned boundary word number from 0 to 2, the phrase table size increases more than 7 times. This is the result of very low recall of word alignments, consequently the estimated phrase table $PP_2(\mathbf{A})$ has very low accuracy. Union suffers from the opposite problem: many incorrect word links prevent good phrase pairs from being extracted.

The two heuristic methods and our proposed optimization approaches achieve somewhat a balance between I and U. By comparing size of $PP_0(\mathbf{A})$ (3rd column), optimization methods are able to identify much more phrase pairs with similar size of alignment set. This confirms that the new method is indeed moving to the desired direction of extracting as many accurate (all boundary words should be aligned) phrase pairs as possible. We still notice that ratio of $|PP_2(\mathbf{A})|$ and $|PP_0(\mathbf{A})|$ (the last column) is high. We suspect that the ratio of this two phrase table size might somewhat be indicative of the phrase table accuracy, which is hard to estimate without manual annotation though.

Method	$\frac{\lvert\mathbf{A}\rvert}{\lvert\mathbf{A}_U\rvert}$	$\lvert PP_0\rvert$	$\lvert PP_1\rvert$	$\lvert PP_2\rvert$	$\frac{\lvert PP_2\rvert}{\lvert PP_0\rvert}$
I	45%	424	2047	3658	8.63
U	100%	354	555	578	1.63
H	78%	538	1225	1519	2.82
GD	82%	499	1081	1484	2.97
OS	84%	592	1110	1210	2.04
OE	78%	659	1359	1615	2.45
HF	95%	427	670	697	1.63
GDF	97%	412	647	673	1.63
OSF	89%	484	752	781	1.61
OEF	89%	476	739	768	1.61

Table 1: Statistics of word alignment set and the resulting phrase table size (number of entries in thousand (K)) with different combination methods

3.2 Translation Results

The ultimate goal of word alignment combination is for building translation system. The quality of resulting phrase tables is measured by automatic translation metric. We have one dev set (1430 sentences with 11483 running words), test set 1 (1390 sentences with 10334 running words) and test set 2 (417 sentences with 4239 running words). The dev set and test set 1 are part of all available Farsi-

English parallel corpus. They are holdout from training data as tuning and testing. The test set 2 is the standard NIST offline evaluation set, where 4 references are available for each sentence. The dev and test set 1 are much closer to the training set than the standard test set 2. We tune all feature weights automatically (Och, 2003) to maximize the BLEU (Papineni et al., 2002) score on the dev set.

Table 2 shows BLEU score of different combination methods on all three sets. Union performs much worse on the dev and test1 than intersection, while intersection achieved the same performance on test2 as union but with more than 6 times of phrase table size. Grow-diagonal (GD) has more than 1 bleu point on test2 than intersection but with less than half of phrase table size. The proposed new method OE is consistently better than both heuristic methods GD and H, with more than 1 point on dev/teset1 and 0.7 point on test2. Comparing the last group to the middle one, we can see the effect of the 'final' operation on all four methods. Tabel 1 shows that after applying the final operation, phrase table size is cut into half. When evaluated with automatic translation metric, all four methods generally perform much worse on dev and test1 that are close to training data, but better on NIST standard test2. We observe half BLEU point improvement for optimization method but marginal gain for heuristic-based approaches. This suggest that the phrase table accuracy get improved with the final operation. Optimization method directly tries to maximize the number of phrase pairs that can be extracted. We observe that it (OEF) is able to find more than 14% more phrase pairs than heuristic methods and achieve 1 BLEU point gain than the best heuristic method (GDF).

Method	dev	test1	test2
I	0.396	0.308	0.348
U	0.341	0.294	0.348
H	0.400	0.314	0.341
GD	0.391	0.314	0.360
OS	0.383	0.316	0.356
OE	0.410	0.329	0.367
HF	0.361	0.297	0.343
GDF	0.361	0.301	**0.362**
OSF	0.372	0.305	0.361
OEF	0.370	0.306	**0.372**

Table 2: Translation results (BLEU score) with phrase tables trained with different word alignment combination methods

4 Conclusions

We presented a simple yet effective method for word alignment symmetrization and combination in general. The problem is formulated as an optimization with greedy search driven by an effectiveness function, which can be customized directly to maximum benefit for intended applications such as phrase table training or synchronized grammar extraction in machine translation. Experimental results demonstrated consistent better BLEU scores than the best heuristic method. The optimization process can better maintain accuracy while improving coverage.

The algorithm is generic and leaves much space for variations. For instance, designing a better effectiveness function g, or considering a soft link with some probability rather than binary 0/1 connection would potentially be opportunities for further improvement. On the other hand, the search space of current algorithm is limited by the pool of candidate set, it is possible to suggest new links while driven by the target function.

Acknowledgments We thank the DARPA TransTac program for funding and the anonymous reviewers for their constructive suggestions.

References

N. F. Ayan. 2005. *Combining Linguistic and Machine Learning Techniques for Word Alignment Improvement*. Ph.D. thesis, University of Maryland, College Park, November.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19:263–312.

S. F. Chen and J. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proc. of ACL*, pages 310–318.

P. Koehn, F. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 48–54.

P. Liang, B. Taskar, and D. Klein. 2006. Alignment by agreement. In *Proc. of HLT-NAACL*, pages 104–111.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

R. Zens, E. Matusov, and H. Ney. 2004. Improved word alignment using a symmetric lexicon model. In *Proc. of COLING*, pages 36–42.

Bridging Morpho-Syntactic Gap between Source and Target Sentences for English-Korean Statistical Machine Translation

Gumwon Hong, Seung-Wook Lee and Hae-Chang Rim
Department of Computer Science & Engineering
Korea University
Seoul 136-713, Korea
{gwhong,swlee,rim}@nlp.korea.ac.kr

Abstract

Often, Statistical Machine Translation (SMT) between English and Korean suffers from null alignment. Previous studies have attempted to resolve this problem by removing unnecessary function words, or by reordering source sentences. However, the removal of function words can cause a serious loss in information. In this paper, we present a possible method of bridging the morpho-syntactic gap for English-Korean SMT. In particular, the proposed method tries to transform a source sentence by inserting pseudo words, and by reordering the sentence in such a way that both sentences have a similar length and word order. The proposed method achieves 2.4 increase in BLEU score over baseline phrase-based system.

1 Introduction

Phrase-based SMT models have performed reasonably well on languages where the syntactic structures are very similar, including languages such as French and English. However, Collins et al. (2005) demonstrated that phrase-based models have limited potential when applied to languages that have a relatively different word order; such is the case between German and English. They proposed a clause restructuring method for reordering German sentences in order to resemble the order of English sentences. By modifying the source sentence structure into the target sentence structure, they argued that they could solve the decoding problem by use of completely monotonic translation.

The translation from English to Korean can be more difficult than the translation of other language pairs for the following reasons: First, Korean is *language isolate*: that is, it has little genealogical relations with other natural languages.[1] Second, the word order in Korean is relatively free because the functional morphemes, case particles and word endings, play the role as a grammatical information marker. Thus, the functional morphemes, rather than the word order, determine whether a word is a subject or an object. Third, Korean is an agglutinative language, in which a word is generally composed of at least one content morpheme and zero or more functional morphemes. Some Korean words are highly synthetic with complex inflections, and this phenomenon produces a very large vocabulary and causes data-sparseness in performing word-based alignment. To mitigate this problem, many systems tokenize Korean sentences by the morpheme unit before training and decoding the sentences.

When analyzing English-Korean translation with MOSES (Koehn et al., 2007), we found high ratio of null alignment. In figure 1, '은(*eun*)', '의(*eui*)', '하(*ha*)', 'ㄴ(*n*)', '지(*ji*)' and '는다(*neunda*)' are not linked to any word in the English sentence. In many cases, these words are function words that are attached to preceding content words. Sometimes they can be linked (incorrectly) to their head's corresponding words, or they can be linked to totally different words with respect to their meaning.

In the preliminary experiment using GIZA++ (Och and Ney, 2003) with grow-diag-final heuristic, we found that about 25% of words in Korean sentences and 21% of English sentences fail to align. This null alignment ratio is relatively high in comparison to the French-English alignment, in which about 9% of French sentences and 6% of English sentences are not aligned. Due to this null alignment, the estimation of translation probabilities for Korean function words may be incomplete; a system would perform mainly based

[1] Some may consider it an Altaic language family.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 233–236,
Suntec, Singapore, 4 August 2009.

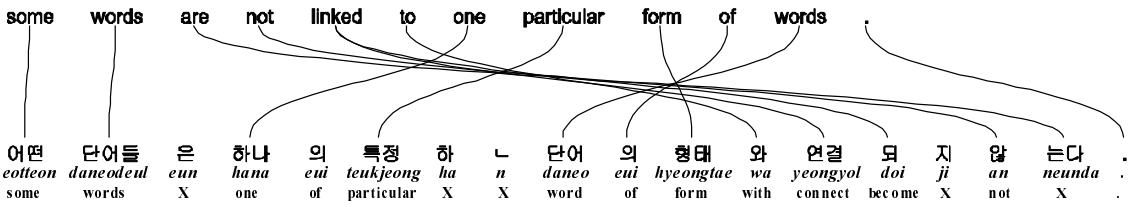


Figure 1: An example of null alignment

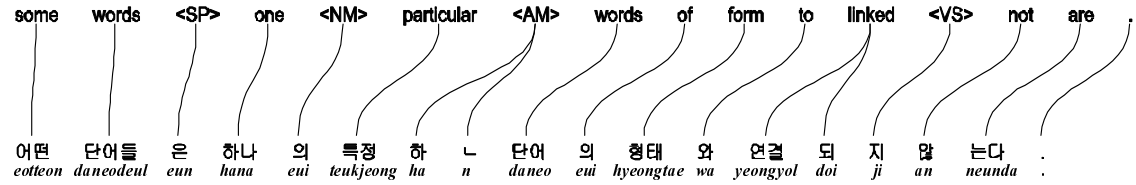


Figure 2: An example of ideal alignment

on content-words, which can deteriorate the performance of candidate generation during decoding. Also, without generating appropriate function words, the quality of the translation will undoubtedly degrade.

In this paper, we present a preprocessing method for both training and decoding in English-Korean SMT. In particular, we transform a source language sentence by inserting pseudo words and syntactically reordering it to form a target sentence structure in hopes of reducing the morpho-syntactic discrepancies between two languages. Ultimately, we expect an ideal alignment, as shown in Figure 2. Our results show that the combined pseudo word insertion and syntactic reordering method reduces null alignment ratio and makes both sentences have similar length. We report results showing that the proposed method can improve the translation quality.

2 Pseudo Word Insertion

Lee et al. (2006) find that function words in Korean sentences are not aligned to any English words, and can simply and easily be removed by referring to their POS information. The unaligned words are case particles, final endings, and auxiliary particles, and they call these words "untranslatable words".

The method can be effective for Korean-English SMT where target language does not have corresponding function words, but it has a limitation in application to the English-Korean SMT because removing functional morphemes can cause a serious loss in information. Technically, the function words they ignored are not 'untranslatable' but are 'unalignable'. Therefore, instead of removing the function words, we decide to insert some *pseudo words* into an English sentence in order to align them with potential Korean function words and make the length of both sentences similar.

To insert the pseudo words, we need to decide: (1) the kinds of words to insert, and (2) the location to insert the words. Because we expect that a pseudo word corresponds to any Korean function word which decides a syntactic role of its head, it is reasonable to utilize a dependency relation of English. Thus, given an English sentence, the candidate pseudo words are generated by the following methods: First, we parse the English sentence using Stanford dependency parser (de Marneffe et al., 2006). Then, we select appropriate typed dependency relations between pairs of words which are able to generate Korean function words. We found that 21 out of 48 dependency relations can be directly used as pseudo words. Among them, some relations provide very strong cue of case particles when inserted as pseudo words.

For example, from the following sentence, we can select as pseudo words a subjective particle <NS> and an objective particle <DO>, and insert them after the corresponding dependents *Eugene* and *guitar* respectively.

nominal_**s**ubject(play, Eugene) **d**irect_**o**bject(play, guitar)
Eugene <NS> can 't play the guitar <DO> well .

In a preliminary experiment on word alignment,

nominal_subject	는(*neun*), null, 이(*i*)
direct_object	을(*eul*), null, 를(*reul*)
clausal_subject	는(*neun*), null, 이(*i*)
temporal_modifier	에(*neun*), null, 오늘(*oneul*)
adj_complement	null, 아(*ah*), 하(*ha*)
agent	null, 에(*e*), 가(*ga*)
numeric_modifier	null, 의(*eui*), 개(*gae*)
adj_modifier	null, 에(*e*), 가(*ga*)
particle_modifier	null, ㄴ(*n*), 되(*doe*)

Figure 3: Selected dependency relations and their aligned function words in training data (shown the top 3 results in descending order of alignment probability)

we observe that inserting too many pseudo words can, on the contrary, increase null alignment of English sentence. Thus we filtered some pseudo words according to their respective null alignment probabilities. Figure 3 shows the top 9 selected dependency relations (actually used in the experiment) and the aligned Korean function words.

3 Syntactic Reordering

Many approaches use syntactic reordering in the preprocessing step for SMT systems (Collins et al., 2005; Xia and McCord, 2004; Zwarts and Dras, 2007). Some reordering approaches have given significant improvements in performance for translation from French to English (Xia and McCord, 2004) and from German to English (Collins et al., 2005). However, on the contrary, Lee et al. (2006) reported that the reordering of Korean for Korean-English translation degraded the performance. They presumed that the performance decrease might come from low parsing performance for conversational domain.

We believe that it is very important to consider the structural properties of Korean for reordering English sentences. Though the word order of a Korean sentence is relatively free, Korean generally observes the SOV word order, and it is a head-final language. Consequently, an object precedes a predicate, and all dependents precede their heads.

We use both a structured parse tree and dependency relations to extract following reordering rules.

- ***Verb final***: In any verb phrase, move verbal head to the end of the phrase. Infinitive verbs or verb particles are moved together.

```
He (likes ((to play) (the piano)))  (1)
He (likes ((the piano) (to play)))  (2)
He (((the piano) (to play)) likes)  (3)
```

- ***Adjective final***: In adjective phrase, move adjective head to the end of the phrase especially if followed by PP or S/SBAR.

```
It is ((difficult) to reorder)  (1)
It is (to reorder (difficult))  (2)
```

- ***Antecedent final***: In noun phrase containing relative clause, move preceding NP to the end of a relative clause.

```
((rules) that are used for reordering)  (1)
(that are used for reordering (rules))  (2)
```

- ***Negation final***: Move negative markers to directly follow verbal head.

```
(can 't) ((play) the guitar)  (1)
(can 't) (the guitar (play))  (2)
(the guitar (play)) (can 't)  (3)
```

4 Experiments

4.1 Experimental Setup

The baseline of our approach is a statistical phrase-based system which is trained using MOSES (Koehn et al., 2007). We collect bilingual texts from the Web and combine them with the Sejong parallel corpora [2]. About 300K pair of sentences are collected from the major bilingual news broadcasting sites. We also collect around 1M monolingual sentences from the sites to train Korean language models. The best performing language model is 5-gram order with Kneser-Ney smoothing.

For sentence level alignment, we modified the Champollion toolkit for English-Korean pair (Ma, 2006). We randomly selected 5,000 sentence pairs from Sejong corpora, of which 1,500 were used for a tuning set for minimum error rate training, and another 1,500 for development set for analysis experiment. We report testing results on the remaining 2,000 sentence pairs for the evaluation.

Korean sentences are tokenized by the morphological analyzer (Lee and Rim, 2004). For English sentence preprocessing, we use the Stanford parser with output of typed dependency relations. We then applied the pseudo word insertion and four reordering rules described in the previous section to the parse tree of each sentence.

[2]The English-Korean parallel corpora open for research purpose which contain about 60,000 sentence pairs. See http://www.sejong.or.kr/english.php for more information

	BLEU(gain)	Length Ratio
Baseline	18.03(+0.00)	0.78
+PWI only	18.62(+0.59)	0.91
+Reorder only	19.92(+1.89)	0.78
+PWI&Reorder	20.42(+2.39)	0.91

Table 1: BLEU score and sentence length ratio for each method

	Baseline	*+PWI*	*+Reorder*	*+P&R*
src-null	20.5	21.4	19.1	20.9
tgt-null	25.4	22.3	23.4	20.8
all-null	23.3	21.9	21.5	20.8

Table 2: Null alignment ratio (%) for each method (all-null is calculated on the whole training data)

4.2 Experimental Results

The BLEU scores are reported in Table 1. Length ratio indicates the average sentence length ratio between source sentences and target sentences. The largest gain (+2.39) is achieved when the combined pseudo word insertion (*PWI*) and word reordering is performed.

There could be reasons why the proposed approach is effective over baseline approach. Presumably, transforming to similar length and word order contributes to lower the distortion and fertility parameter values. Table 2 analyzes the effect of individual techniques in terms of the null alignment ratio. We discover that the alignment ratio can be a good way to measure the relation between the quality of word alignment and the quality of translation. As shown in Table 2, the BLEU score tends to increase as the all-null ratio decreases. Interestingly, reordering achieves the smallest null alignment ratio for source language.

5 Conclusions

In this paper, we presented a novel approach to preprocessing English-Korean SMT. The morpho-syntactic discrepancy between English and Korean causes a serious null alignment problem.

The main contributions of this paper are the following: 1) we devise a new preprocessing method for English-Korean SMT by transforming a source sentence to be much closer to a target sentence in terms of sentence length and word order. 2) we discover that the proposed method can reduce the null alignment problem, and consequently the null word alignment ratio between two languages can be a good way to measure the quality of translation.

When evaluating the proposed approach using within MOSES, the combined pseudo word insertion and syntactic reordering method outperforms the other methods. The result proves that the proposed method can be used as a useful technique for English-Korean machine translation.

Acknowledgments

This work was supported by Microsoft Research Asia. Any opinions, findings, and conclusions or recommendations expressed above are those of the authors and do not necessarily reflect the views of the sponsor.

References

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proc. of ACL*.

Marie-Catherine de Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of ACL Demonstration session*.

Do-Gil Lee and Hae-Chang Rim. 2004. Part-of-speech tagging considering surface form for an agglutinative language. In *Proc. of ACL*.

Jonghoon Lee, Donghyeon Lee, and Gary Geunbae Lee. 2006. Improving phrase-based korean-english statistical machine translation. In *Proc. of Interspeech-ICSLP*.

Xiaoyi Ma. 2006. Champollion: A robust parallel text sentence aligner. In *Proc. of LREC*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Fei Xia and Michael McCord. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proc. of COLING*.

Simon Zwarts and Mark Dras. 2007. Syntax-based word reordering in phrase-based statistical machine translation: Why does it work? In *Proc. of MT-Summit XI*.

Toward Smaller, Faster, and Better Hierarchical Phrase-based SMT

Mei Yang
Dept. of Electrical Engineering
University of Washington, Seattle, WA, USA
yangmei@u.washington.edu

Jing Zheng
SRI International
Menlo Park, CA, USA
zj@speech.sri.com

Abstract

We investigate the use of Fisher's exact significance test for pruning the translation table of a hierarchical phrase-based statistical machine translation system. In addition to the significance values computed by Fisher's exact test, we introduce compositional properties to classify phrase pairs of same significance values. We also examine the impact of using significance values as a feature in translation models. Experimental results show that 1% to 2% BLEU improvements can be achieved along with substantial model size reduction in an Iraqi/English two-way translation task.

1 Introduction

Phrase-based translation (Koehn et al., 2003) and hierarchical phrase-based translation (Chiang, 2005) are the state of the art in statistical machine translation (SMT) techniques. Both approaches typically employ very large translation tables extracted from word-aligned parallel data, with many entries in the tables never being used in decoding. The redundancy of translation tables is not desirable in real-time applications, e.g., speech-to-speech translation, where speed and memory consumption are often critical concerns. In addition, some translation pairs in a table are generated from training data errors and word alignment noise. Removing those pairs could lead to improved translation quality.

(Johnson et al., 2007) has presented a technique for pruning the phrase table in a phrase-based SMT system using Fisher's exact test. They compute the significance value of each phrase pair and prune the table by deleting phrase pairs with significance values smaller than a threshold. Their experimental results show that the size of the phrase table can be greatly reduced with no significant loss in translation quality.

In this paper, we extend the work in (Johnson et al., 2007) to a hierarchical phrase-based translation model, which is built on synchronous context-free grammars (SCFG). We call an SCFG rule a phrase pair if its right-hand side does not contain a nonterminal, and otherwise a rewrite rule. Our approach applies to both the phrase table and the rule table. To address the problem that many translation pairs share the same significance value from Fisher's exact test, we propose a refined method that combines significance values and compositional properties of surface strings for pruning the phrase table. We also examine the effect of using the significance values as a feature in translation models.

2 Fisher's exact test for translation table pruning

2.1 Significance values by Fisher's exact test

We briefly review the approach for computing the significance value of a translation pair using Fisher's exact test. In Fisher's exact test, the significance of the association of two items is measured by the probability of seeing the number of co-occurrences of the two items being the same as or higher than the one observed in the sample. This probability is referred to as the p-value. Given a parallel corpus consisting of N sentence pairs, the probability of seeing a pair of phrases (or rules) $(\tilde{s}, \tilde{t})$ with the joint frequency $C(\tilde{s}, \tilde{t})$ is given by the hypergeometric distribution

$$P_h(C(\tilde{s},\tilde{t})) = \frac{C(\tilde{s})!(N-C(\tilde{s}))!C(\tilde{t})!(N-C(\tilde{t}))!}{N!C(\tilde{s},\tilde{t})!C(\tilde{s},\neg\tilde{t})!C(\neg\tilde{s},\tilde{t})!C(\neg\tilde{s},\neg\tilde{t})!}$$

where $C(\tilde{s})$ and $C(\tilde{t})$ are the marginal frequencies of $\tilde{s}$ and $\tilde{t}$, respectively. $C(\tilde{s},\neg\tilde{t})$ is the number of sentence pairs that contain $\tilde{s}$ on the source side

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 237–240,
Suntec, Singapore, 4 August 2009.

but do not contain $\tilde{t}$ on the target side, and similar for the definition of $C(\neg\tilde{s}, \tilde{t})$ and $C(\neg\tilde{s}, \neg\tilde{t})$. The p-value is therefore the sum of the probabilities of seeing the two phrases (or rules) occur as often as or more often than $C(\tilde{s}, \tilde{t})$ but with the same marginal frequencies

$$P_v(C(\tilde{s}, \tilde{t})) = \sum_{c=C(\tilde{s},\tilde{t})}^{\infty} P_h(c)$$

In practice, p-values can be very small, and thus negative logarithm p-values are often used instead as the measure of significance. In the rest of this paper, the negative logarithm p-value is referred to as the *significance value*. Therefore, the larger the value, the greater the significance.

2.2 Table pruning with significance values

The basic scheme to prune a translation table is to delete all translation pairs that have significance values smaller than a given threshold.

However, in practice, this pruning scheme does not work well with phrase tables, as many phrase pairs receive the same significance values. In particular, many phrase pairs in the phrase table have joint and both marginal frequencies all equal to 1. Such phrase pairs are referred to as *triple-1* pairs. It can be shown that the significance value of triple-1 phrase pairs is $log(N)$. Given a threshold, triple-1 phrase pairs either all remain in the phrase table or are discarded entirely.

To look closer at the problem, Figure 1 shows two example tables with their percentages of phrase pairs that have higher, equal, or lower significance values than $log(N)$. When the threshold is smaller than $log(N)$, as many as 35% of the phrase pairs can be deleted. When the threshold is greater than $log(N)$, at least 90% of the phrase pairs will be discarded. There is no threshold that prunes the table in the range of 35% to 90%. One may think that it is right to delete all triple-1 phrase pairs as they occur only once in the parallel corpus. However, it has been shown in (Moore, 2004) that when a large number of singleton-singleton pairs, such as triple-1 phrase pairs, are observed, most of them are not due to chance. In other words, most triple-1 phrase pairs are significant and it is likely that the translation quality will decline if all of them are discarded. Therefore, using significance values alone cannot completely resolve the problem of phrase table pruning. To further discriminate phrase pairs

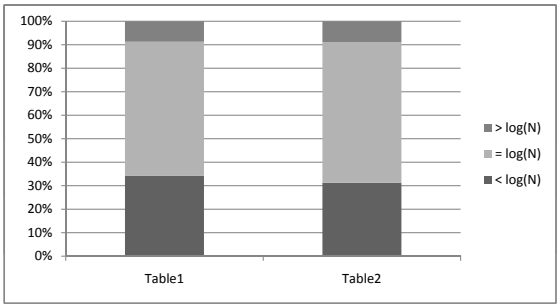


Figure 1: Percentages of phrase pairs with higher, equal, and lower significance values than $log(N)$.

of the same significance values, particularly the triple-1 phrase pairs, more information is needed.

The Fisher's exact test does not consider the surface string in phrase pairs. Intuitively, some phrase pairs are less important if they can be constructed by other phrase pairs in the decoding phase, while other phrase pairs that involve complex syntactic structures are usually difficult to construct and thus become more important. This intuition inspires us to explore the compositional property of a phrase pair as an additional factor. More formally, we define the compositional property of a phrase pair as the capability of decomposing into subphrase pairs. If a phrase pair $(\tilde{s}, \tilde{t})$ can be decomposed into K subphrase pairs $(\tilde{s}_k, \tilde{t}_k)$ already in the phrase table such that

$$\tilde{s} = \tilde{s}_1\tilde{s}_2 \ldots \tilde{s}_K$$
$$\tilde{t} = \tilde{t}_1\tilde{t}_2 \ldots \tilde{t}_K$$

then this phrase pair is compositional; otherwise it is noncompositional. Our intuition suggests that noncompositional phrase pairs are more important as they cannot be generated by concatenating other phrase pairs in order in the decoding phase. This leads to a refined scheme for pruning the phrase table, in which a phrase pair is discarded when it has a significance value smaller than the threshold and it is not a noncompositional triple-1 phrase pair. The definition of the compositional property does not allow re-ordering. If re-ordering is allowed, all phrase pairs will be compositional as they can always be decomposed into pairs of single words.

In the rule table, however, the percentage of triple-1 pairs is much smaller, typically less than 10%. This is because rules are less sparse than phrases in general, as they are extracted with a shorter length limit, and have nonterminals that match any span of words. Therefore, the basic pruning scheme works well with rule tables.

3 Experiment

3.1 Hierarchical phrase-based SMT system

Our hierarchical phrase-based SMT system translates from Iraqi Arabic (IA) to English (EN) and vice versa. The training corpus consists of 722K aligned Iraqi and English sentence pairs and has 5.0M and 6.7M words on the Iraqi and English sides, respectively. A held-out set with 18K Iraqi and 19K English words is used for parameter tuning and system comparison. The test set is the TRANSTAC June08 offline evaluation data with 7.4K Iraqi and 10K English words, and the translation quality is evaluated by case-insensitive BLEU with four references.

3.2 Results on translation table pruning

For each of the two translation directions IA-to-EN and EN-to-IA, we pruned the translation tables as below, where α represents the significance value of triple-1 pairs and ε is a small positive number. Phrase table *PTABLE3* is obtained using the refined pruning scheme, and others are obtained using the basic scheme. Figure 2 shows the percentages of translation pairs in these tables.

- *PTABLE0*: phrase table of full size without pruning.
- *PTABLE1*: pruned phrase table using the threshold $\alpha - \varepsilon$ and thus all triple-1 phrase pairs remain.
- *PTABLE2*: pruned phrase table using the threshold $\alpha + \varepsilon$ and thus all triple-1 phrase pairs are discarded.
- *PTABLE3*: pruned phrase table using the threshold $\alpha + \varepsilon$ and the refined pruning scheme. All but noncompositional triple-1 phrase pairs are discarded.
- *RTABLE0*: rule table of full size without pruning.
- *RTABLE1*: pruned rule table using the threshold $\alpha + \varepsilon$.

Since a hierarchical phrase-based SMT system requires a phrase table and a rule table at the same time, performance of different combinations of phrase and rule tables is evaluated. The baseline system will be the one using the full-size tables of *PTABLE0* and *RTABLE0*. Tables 2 and 3 show the BLEU scores for each combination in each direction, with the best score in bold.

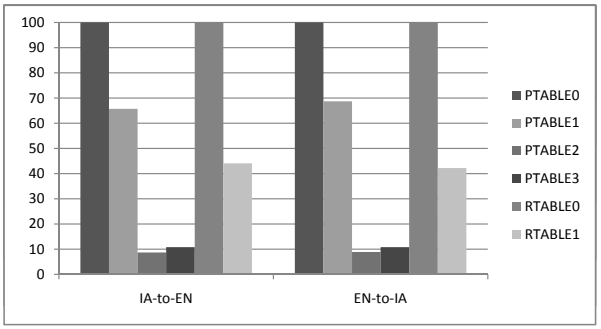


Figure 2: The percentages of translation pairs in phrase and rule tables.

It can be seen that pruning leads to a substantial reduction in the number of translation pairs. As long phrases are more frequently pruned than short phrases, the actual memory saving is even more significant. It is surprising to see that using pruned tables improves the BLEU scores in many cases, probably because a smaller translation table generalizes better on an unseen test set, and some translation pairs created by erroneous training data are dropped. Table 1 shows two examples of discarded phrase pairs and their frequencies. Both of them are incorrect due to human translation errors.

We note that using the pruned rule table *RTABLE1* is very effective and improved BLEU in most cases except when used with *PTABLE0* in the direction EN-to-IA. Although using the pruned phrase tables had mixed effect, *PTABLE3*, which is obtained through the refined pruning scheme, outperformed others in all cases. This confirms the hypothesis that noncompositional phrase pairs are important and thus suggests that the proposed compositional property is a useful measure of phrase pair quality. Overall, the best results are achieved by using the combination of *PTABLE3* and *RTABLE1*, which gave improvement of 1% to 2% BLEU over the baseline systems. Meanwhile, this combination is also twice faster than the baseline system in decoding.

3.3 Results on using significance values as a feature

The p-value of each translation pair can be used as a feature in the log-linear translation model, to penalize those less significant phrase pairs and rewrite rules. Since component feature values cannot be zero, a small positive number was added to p-values to avoid infinite log value. The results of using p-values as a feature with different combinations of phrase and rule tables are shown in

Iraqi Arabic phrase	English phrase in data	Correct English phrase	Frequencies
إحنا خمسة	there are **four** of us	there are **five** of us	1, 29, 1
الشباب ثلاثة أو أربع	young men three **of** four	young men three **or** four	1, 1, 1

Table 1: Examples of pruned phrase pairs and their frequencies $C(\tilde{s},\tilde{t})$, $C(\tilde{s})$, and $C(\tilde{t})$.

	RTABLE0	RTABLE1
PTABLE0	47.38	48.40
PTABLE1	47.05	48.45
PTABLE2	47.50	48.70
PTABLE3	47.81	**49.43**

Table 2: BLEU scores of IA-to-EN systems using different combinations of phrase and rule tables.

	RTABLE0	RTABLE1
PTABLE0	29.92	29.05
PTABLE1	29.62	30.60
PTABLE2	29.87	30.57
PTABLE3	30.62	**31.27**

Table 3: BLEU scores of EN-to-IA systems using different combinations of phrase and rule tables.

	RTABLE0	RTABLE1
PTABLE0	47.72	47.96
PTABLE1	46.69	48.75
PTABLE2	47.90	48.48
PTABLE3	47.59	**49.50**

Table 4: BLEU scores of IA-to-EN systems using the feature of p-values in different combinations.

	RTABLE0	RTABLE1
PTABLE0	29.33	30.44
PTABLE1	30.28	30.99
PTABLE2	30.38	31.44
PTABLE3	30.74	**31.64**

Table 5: BLEU scores of EN-to-IA systems using the feature of p-values in different combinations.

Tables 4 and 5. We can see that the results obtained by using the full rule table with the feature of p-values (the columns of *RTABLE0* in Tables 4 and 5) are much worse than those obtained by using the pruned rule table without the feature of p-values (the columns of *RTABLE1* in Tables 2 and 3). This suggests that the use of significance values as a feature in translation models is not as efficient as the use in translation table pruning. Modest improvement was observed in the direction EN-to-IA when both pruning and the feature of p-values are used (compare the columns of *RTABLE1* in Tables 3 and 5) but not in the direction IA-to-EN. Again, the best results are achieved by using the combination of *PTABLE3* and *RTABLE1*.

4 Conclusion

The translation quality and speed of a hierarchical phrase-based SMT system can be improved by aggressive pruning of translation tables. Our proposed pruning scheme, which exploits both significance values and compositional properties, achieved the best translation quality and gave improvements of 1% to 2% on BLEU when compared to the baseline system with full-size tables. The use of significance values in translation table pruning and in translation models as a feature has a different effect: the former led to significant improvement, while the latter achieved only modest or no improvement on translation quality.

5 Acknowledgements

Many thanks to Kristin Precoda and Andreas Kathol for valuable discussion. This work is supported by DARPA, under subcontract 55-000916 to UW under prime contract NBCHD040058 to SRI International.

References

Philipp Koehn, Franz J. Och and Daniel Marcu. 2003. *Statistical phrase-based translation*. Proceedings of HLT-NAACL, 48-54, Edmonton, Canada.

David Chiang. 2005. *A hierarchical phrase-based model for statistical machine translation*. Proceedings of ACL, 263-270, Ann Arbor, Michigan, USA.

J Howard Johnson, Joel Martin, George Foster and Roland Kuhn. 2007. *Improving Translation Quality by Discarding Most of the Phrasetable*. Proceedings of EMNLP-CoNLL, 967-975, Prague, Czech Republic.

Robert C. Moore. 2004. *On Log-Likelihood-Ratios and the Significance of Rare Events*. Proceedings of EMNLP, 333-340, Barcelona, Spain

Handling phrase reorderings for machine translation

Yizhao Ni, Craig J. Saunders,* Sandor Szedmak and Mahesan Niranjan
ISIS Group
School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ
United Kingdom
yn05r@ecs.soton.ac.uk, craig.saunders@xrce.xerox.com,
{ss03v,mn}@ecs.soton.ac.uk

Abstract

We propose a distance phrase reordering model (DPR) for statistical machine translation (SMT), where the aim is to capture phrase reorderings using a structure learning framework. On both the reordering classification and a Chinese-to-English translation task, we show improved performance over a baseline SMT system.

1 Introduction

Word or phrase reordering is a common problem in bilingual translations arising from different grammatical structures. For example, in Chinese the expression of the date follows "Year/Month/Date", while when translated into English, "Month/Date/Year" is often the correct grammar. In general, the fluency of machine translations can be greatly improved by obtaining the correct word order in the target language.

As the reordering problem is computationally expensive, a word distance-based reordering model is commonly used among SMT decoders (Koehn, 2004), in which the costs of phrase movements are linearly proportional to the reordering distance. Although this model is simple and efficient, the content independence makes it difficult to capture many distant phrase reordering caused by the grammar. To tackle the problem, (Koehn et al., 2005) developed a *lexicalized reordering model* that attempted to learn the phrase reordering based on content. The model learns the local orientation (e.g. "monotone" order or "switching" order) probabilities for each bilingual phrase pair using Maximum Likelihood Estimation (MLE). These orientation probabilities are then integrated into an SMT decoder to help finding a Viterbi–best local orientation sequence. Improvements by this model have been reported in (Koehn et al., 2005). However, the amount of the training data for each bilingual phrase is so small that the model usually suffers from the data sparseness problem. Adopting the idea of predicting the orientation, (Zens and Ney, 2006) started exploiting the context and grammar which may relate to phrase reorderings. In general, a *Maximum Entropy* (ME) framework is utilized and the feature parameters are tuned by a discriminative model. However, the training times for ME models are usually relatively high, especially when the output classes (i.e. phrase reordering orientations) increase.

Alternative to the ME framework, we propose using a classification scheme here for phrase reorderings and employs a structure learning framework. Our results confirm that this distance phrase reordering model (DPR) can lead to improved performance with a reasonable time efficiency.

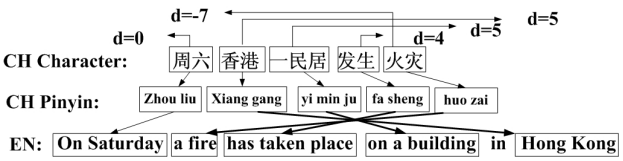


Figure 1: The phrase reordering distance d.

2 Distance phrase reordering (DPR)

We adopt a discriminative model to capture the frequent distant reordering which we call *distance phrase reordering*. An ideal model would consider every position as a class and predict the position of the next phrase, although in practice we must consider a limited set of classes (denoted as Ω). Using the reordering distance d (see Figure 1) as defined by (Koehn et al., 2005), we extend the two class model in (Xiong et al., 2006) to multiple classes (e.g. three–class setup $\Omega = \{d < 0, d = 0, d > 0\}$; or five–class setup $\Omega = \{d \leq -5, -5 < d < 0, d = 0, 0 < d < 5, d \geq 5\}$). Note that the more

*the author's new address: Xerox Research Centre Europe 6, Chemin de Maupertuis, 38240 Meylan France.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 241–244,
Suntec, Singapore, 4 August 2009.

classes it has, the closer it is to the ideal model, but the smaller amount of training samples it would receive for each class.

2.1 Reordering Probability model and training algorithm

Given a (source, target) phrase pair $(\bar{f}_j, \bar{e}_i)$ with $\bar{f}_j = [f_{j_l}, \ldots, f_{j_r}]$ and $\bar{e}_i = [e_{i_l}, \ldots, e_{i_r}]$, the distance phrase reordering probability has the form

$$p(o|\bar{f}_j, \bar{e}_i) := \frac{h\big(\mathbf{w}_o^T \phi(\bar{f}_j, \bar{e}_i)\big)}{\sum_{o' \in \Omega} h\big(\mathbf{w}_{o'}^T \phi(\bar{f}_j, \bar{e}_i)\big)} \qquad (1)$$

where $\mathbf{w}_o = [w_{o,0}, \ldots, w_{o,dim(\phi)}]^T$ is the weight vector measuring features' contribution to an orientation $o \in \Omega$, ϕ is the feature vector and h is a pre-defined monotonic function. As the reordering orientations tend to be interdependent, learning $\{\mathbf{w}_o\}_{o \in \Omega}$ is more than a multi–class classification problem. Take the five–class setup for example, if an example in class $d \leq -5$ is classified in class $-5 < d < 5$, intuitively the loss should be smaller than when it is classified in class $d > 5$. The output (orientation) domain has an inherent structure and the model should respect it. Hence, we utilize the structure learning framework proposed in (Taskar et al., 2003) which is equivalent to minimising the sum of the classification errors

$$\min_{\mathbf{w}} \frac{1}{N} \sum_{n=1}^{N} \rho(o, \bar{f}_j^n, \bar{e}_i^n, \mathbf{w}) + \frac{\lambda}{2} \|\mathbf{w}\|^2 \qquad (2)$$

where $\lambda \geq 0$ is a regularisation parameter,

$$\rho(o, \bar{f}_j, \bar{e}_i, \mathbf{w}) = \max\{0, \max_{o' \neq o}[\triangle(o, o') + \mathbf{w}_{o'}^T \phi(\bar{f}_j, \bar{e}_i)] - \mathbf{w}_o^T \phi(\bar{f}_j, \bar{e}_i)\}$$

is a structured margin loss function with

$$\triangle(o, o') = \begin{cases} 0 & \text{if } o = o' \\ 0.5 & \text{if } o \text{ and } o' \text{ are close in } \Omega \\ 1 & \text{else} \end{cases}$$

measuring the distance between pseudo orientation o' and the true one o. Theoretically, this loss requires that orientation o' which are "far away" from the true one o must be classified with a large margin while nearby candidates are allowed to be classified with a smaller margin. At training time, we used a perceptron–based structure learning (PSL) algorithm to learn $\{\mathbf{w}_o\}_{o \in \Omega}$ which is shown in Table 1.

Input: The samples $\{o, \phi(\bar{f}_j, \bar{e}_i)\}_{n=1}^N$, step size η
Initialization: $k = 0$; $\mathbf{w}_{o,k} = \mathbf{0} \quad \forall o \in \Omega$;
Repeat
  for $n = 1, 2, \ldots, N$ **do**
    for $o' \neq o$ **get**
      $V = \max_{o'} \{\triangle(o, o') + \mathbf{w}_{o',k}^T \phi(\bar{f}_j, \bar{e}_i)\}$
      $o^* = \arg\max_{o'} \{\triangle(o, o') + \mathbf{w}_{o',k}^T \phi(\bar{f}_j, \bar{e}_i)\}$
    if $w_{o,k}^T \phi(\bar{f}_j, \bar{e}_i) < V$ **then**
      $\mathbf{w}_{o,k+1} = \mathbf{w}_{o,k} + \eta \phi(\bar{f}_j, \bar{e}_i)$
      $\mathbf{w}_{o^*,k+1} = \mathbf{w}_{o^*,k} - \eta \phi(\bar{f}_j, \bar{e}_i)$
  $k = k + 1$
until converge
Output: $\mathbf{w}_{o,k+1} \quad \forall o \in \Omega$

Table 1: Perceptron-based structure learning.

2.1.1 Feature Extraction and Application

Following (Zens and Ney, 2006), we consider different kinds of information extracted from the phrase environment (see Table 2), where given a sequence s (e.g. $s = [f_{j_l - z}, \ldots, f_{j_l}]$), the features selected are $\phi_u(s_p^{|u|}) = \delta(s_p^{|u|}, u)$, with the indicator function $\delta(\cdot, \cdot)$, $p = \{j_l - z, \ldots, j_r + z\}$ and string $s_p^{|u|} = [f_p, \ldots, f_{p+|u|}]$. Hence, the phrase features are distinguished by both the content u and its start position p. For example, the left side context features for phrase pair (xiang gang, Hong Kong) in Figure 1 are $\{\delta(s_0^1, \text{"zhou"}), \delta(s_1^1, \text{"liu"}), \delta(s_0^2, \text{"zhou liu"})\}$. As required by the algorithm, we then *normalise* the feature vector $\bar{\phi}_t = \frac{\phi_t}{\|\phi\|}$.

To train the DPR model, the training samples $\{(\bar{f}_j^n, \bar{e}_i^n)\}_{n=1}^N$ are extracted following the phrase pair extraction procedure in (Koehn et al., 2005) and form the sample pool, where the instances having the same source phrase $\bar{f}_j$ are considered to be from the same cluster. A sub-DPR model is then trained for each cluster using the PSL algorithm. During the decoding, the DPR model finds the corresponding sub-DPR model for a source phrase $\bar{f}_j$ and generates the reordering probability for each orientation class using equation (1).

3 Experiments

Experiments used the Hong Kong Laws corpus[1] (Chinese-to-English), where sentences of lengths between 1 and 100 words were extracted and the ratio of source/target lengths was no more than $2:1$. The training and test sizes are $50,290$ and $1,000$ respectively.

[1]This bilingual Chinese-English corpus consists of mainly legal and documentary texts from Hong Kong. The corpus is aligned at the sentence level which are collected and revised manually by the author. The full corpus will be released soon.

	Features for source phrase $\bar{f}_j$	Features for target phrase $\bar{e}_i$
Context	Source word n–grams within a window (length z) around the phrase edge $[j_l]$ and $[j_r]$	Target word n–grams of the phrase $[e_{i_l}, \ldots, e_{i_r}]$
Syntactic	Source word class tag n-grams within a window (length z) around the phrase edge $[j_l]$ and $[j_r]$	Target word class tag n-grams of the phrase $[e_{i_l}, \ldots, e_{i_r}]$

Table 2: The environment for the feature extraction. The word class tags are provided by MOSES.

3.1 Classification Experiments

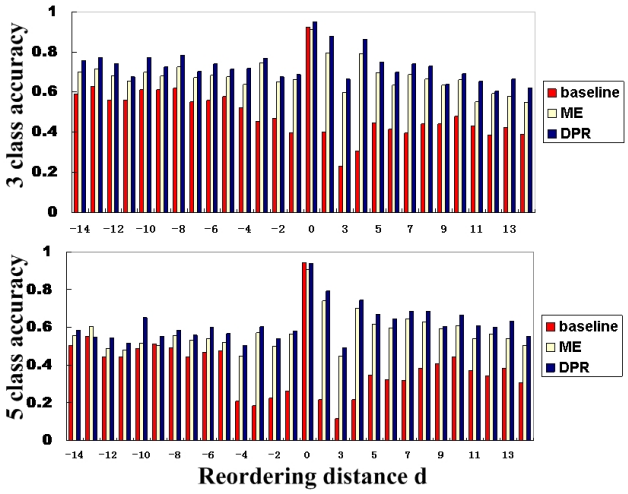


Figure 2: Classification results with respect to d.

We used GIZA++ to produce alignments, enabling us to compare using a DPR model against a baseline lexicalized reordering model (Koehn et al., 2005) that uses MLE orientation prediction and a discriminative model (Zens and Ney, 2006) that utilizes an ME framework. Two orientation classification tasks are carried out: one with three–class setup and one with five–class setup. We discarded points that had long distance reordering ($|d| > 15$) to avoid some alignment errors cause by GIZA++ (representing less than 5% of the data). This resulted in data sizes shown in Table 3. The classification performance is measured by an overall precision across all classes and the class-specific F1 measures and the experiments are are repeated three times to asses variance.

Table 4 depicts the classification results obtained, where we observed consistent improvements for the DPR model over the baseline and the ME models. When the number of classes (orientations) increases, the average relative improvements of DPR for the switching classes (i.e. $d \neq 0$) increase from 41.6% to 83.2% over the baseline and from 7.8% to 14.2% over the ME model, which implies a potential benefit of structure learning. Figure 2 further demonstrate the average accuracy for each reordering distance d. It shows that even for long distance reordering, the DPR model still performs well, while the MLE baseline usually performs badly (more than half examples are classified incorrectly). With so many classification errors, the effect of this baseline in an SMT system is in doubt, even with a powerful language model. At training time, training a DPR model is much faster than training an ME model (both algorithms are coded in Python), especially when the number of classes increase. This is because the generative iterative scaling algorithm of an ME model requires going through all examples twice at each round: one is for updating the conditional distributions $p(o|\bar{f}_j, \bar{e}_i)$ and the other is for updating $\{\mathbf{w}_o\}_{o\in\Omega}$. Alternatively, the PSL algorithm only goes through all examples once at each round, making it faster and more applicable for larger data sets.

3.2 Translation experiments

We now test the effect of the DPR model in an MT system, using MOSES (Koehn et al., 2005) as a baseline system. To keep the comparison fair, our MT system just replaces MOSES's reordering models with DPR while sharing all other models (i.e. phrase translation probability model, 4-gram language model (A. Stolcke, 2002) and beam search decoder). As in classification experiments the three-class setup shows better results in switching classes, we use this setup in DPR. In detail, all consistent phrases are extracted from the training sentence pairs and form the sample pool. The three-class DPR model is then trained by the PSL algorithm and the function $h(z) = \exp(z)$ is applied to equation (1) to transform the prediction scores. Contrasting the direct use of the reordering probabilities used in (Zens and Ney, 2006), we utilize the probabilities to adjust the word distance–based reordering cost, where the reordering cost of a sentence is computed as $P_o(\mathbf{f}, \mathbf{e}) =$

Settings	three–class setup			five–class setup				
Classes	$d < 0$	$d = 0$	$d > 0$	$d \leq -5$	$-5 < d < 0$	$d = 0$	$0 < d < 5$	$d \geq 5$
Train	181,583	755,854	181,279	82,677	98,907	755,854	64,881	116,398
Test	5,025	21,106	5,075	2,239	2,786	21,120	1,447	3,629

Table 3: Data statistics for the classification experiments.

System	three–class setup task				
	Precision	$d < 0$	$d = 0$	$d > 0$	Training time (hours)
Lexicalized	77.1 ± 0.1	55.7 ± 0.1	86.5 ± 0.1	49.2 ± 0.3	1.0
ME	83.7 ± 0.3	67.9 ± 0.3	90.8 ± 0.3	69.2 ± 0.1	58.6
DPR	$\mathbf{86.7} \pm 0.1$	$\mathbf{73.3} \pm 0.1$	$\mathbf{92.5} \pm 0.2$	$\mathbf{74.6} \pm 0.5$	27.0

System	five–class setup task						
	Precision	$d \leq -5$	$-5 < d < 0$	$d = 0$	$0 < d < 5$	$d \geq 5$	Training Time (hours)
Lexicalized	74.3 ± 0.1	44.9 ± 0.2	32.0 ± 1.5	86.4 ± 0.1	29.2 ± 1.7	46.2 ± 0.8	1.3
ME	80.0 ± 0.2	52.1 ± 0.1	54.7 ± 0.7	90.4 ± 0.2	63.9 ± 0.1	61.8 ± 0.1	83.6
DPR	$\mathbf{84.6 \pm 0.1}$	$\mathbf{60.0} \pm 0.7$	$\mathbf{61.4} \pm 0.1$	$\mathbf{92.6} \pm 0.2$	$\mathbf{75.4} \pm 0.6$	$\mathbf{68.8} \pm 0.5$	29.2

Table 4: Overall precision and class-specific F1 scores [%] using different number of orientation classes. Bold numbers refer to the best results.

$\exp\{-\sum_m \frac{d_m}{\beta p(o|\bar{f}_{jm}, \bar{e}_{im})}\}$ with tuning parameter β. This distance–sensitive expression is able to fill the deficiency of the three–class setup of DPR and is verified to produce better results. For parameter tuning, minimum-error-rating training (F. J. Och, 2003) is used in both systems. Note that there are 7 parameters needed tuning in MOSES's reordering models, while only 1 requires tuning in DPR. The translation performance is evaluated by four MT measurements used in (Koehn et al., 2005).

Table 5 shows the translation results, where we observe consistent improvements on most evaluations. Indeed both systems produced similar word accuracy, but our MT system does better in phrase reordering and produces more fluent translations.

4 Conclusions and Future work

We have proposed a distance phrase reordering model using a structure learning framework. The classification tasks have shown that DPR is better in capturing the phrase reorderings over the lexicalized reordering model and the ME model. Moreover, compared with ME DPR is much faster and more applicable to larger data sets. Translation experiments carried out on the Chinese-to-English task show that DPR gives more fluent translation results, which verifies its effectiveness. For future work, we aim at improving the prediction accuracy for the five-class setup using a richer feature set before applying it to an MT system, as DPR can be more powerful if it is able to provide more precise phrase position for the decoder. We will also apply DPR on a larger data set to test its performance as well as its time efficiency.

Tasks	Measure	MOSES	DPR
CH–EN	BLEU [%]	44.7 ± 1.2	$\mathbf{47.1} \pm 1.3$
	word accuracy	$\mathbf{76.5} \pm 0.6$	76.1 ± 1.5
	NIST	8.82 ± 0.11	$\mathbf{9.04} \pm 0.26$
	METEOR [%]	66.1 ± 0.8	$\mathbf{66.4} \pm 1.1$

Table 5: Four evaluations for the MT experiments. Bold numbers refer to the best results.

References

P. Koehn. 2004. Pharaoh: a beam search decoder for phrase–based statistical machine translation models. In *Proc. of AMTA 2004*, Washington DC, October.

P. Koehn, A. Axelrod, A. B. Mayne, C. Callison–Burch, M. Osborne and D. Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proc. of IWSLT*, Pittsburgh, PA.

F. J. Och. 2003. SRILM - An Extensible Language Modeling Toolkit. In *Proc. Intl. Conf. Spoken Language Processing*, Colorado, September.

A. Stolcke. 2002. Minimum error rate training in statistical machine translation. In *Proc. ACL*, Japan.

B. Taskar, C. Guestrin, and D.Koller. 2003. Max–margin Markov networks. In *Proc. NIPS*, Vancouver, Canada, December.

D. Xiong, Q. Liu and S. Lin. 2006. Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation. In *Proc. of ACL*, Sydney, July.

R. Zens and H. Ney. 2006. Discriminative Reordering Models for Statistical Machine Translation. In *Proc. of ACL*, pages 55–63, New York City, June.

Syntax is from Mars while Semantics from Venus! Insights from Spectral Analysis of Distributional Similarity Networks

Chris Biemann
Microsoft/Powerset, San Francisco
Chris.Biemann@microsoft.com

Monojit Choudhury
Microsoft Research Lab India
monojitc@microsoft.com

Animesh Mukherjee
Indian Institute of Technology Kharagpur, India
animeshm@cse.iitkgp.ac.in

Abstract

We study the global topology of the syntactic and semantic distributional similarity networks for English through the technique of spectral analysis. We observe that while the syntactic network has a hierarchical structure with strong communities and their mixtures, the semantic network has several tightly knit communities along with a large core without any such well-defined community structure.

1 Introduction

Syntax and semantics are two tightly coupled, yet very different properties of any natural language – as if one is from "Mars" and the other from "Venus". Indeed, this exploratory work shows that the distributional properties of syntax are quite different from those of semantics. *Distributional hypothesis* states that the words that occur in the same contexts tend to have similar meanings (Harris, 1968). Using this hypothesis, one can define a vector space model for words where every word is a point in some n-dimensional space and the distance between them can be interpreted as the inverse of the semantic or syntactic similarity between their corresponding distributional patterns. Usually, the co-occurrence patterns with respect to the function words are used to define the syntactic context, whereas that with respect to the content words define the semantic context. An alternative, but equally popular, visualization of distributional similarity is through graphs or networks, where each word is represented as nodes and weighted edges indicate the extent of distributional similarity between them.

What are the commonalities and differences between the syntactic and semantic distributional patterns of the words of a language? This study is an initial attempt to answer this fundamental and intriguing question, whereby we construct the syntactic and semantic distributional similarity network (DSN) and analyze their spectrum to understand their global topology. We observe that there are significant differences between the two networks: the syntactic network has well-defined hierarchical community structure implying a systematic organization of natural classes and their mixtures (e.g., words which are both nouns and verbs); on the other hand, the semantic network has several isolated clusters or the so called *tightly knit communities* and a core component that lacks a clear community structure. Spectral analysis also reveals the basis of formation of the natural classes or communities within these networks. These observations collectively point towards a well accepted fact that the semantic space of natural languages has extremely high dimension with no clearly observable subspaces, which makes theorizing and engineering harder compared to its syntactic counterpart.

Spectral analysis is the backbone of several techniques, such as multi-dimensional scaling, principle component analysis and latent semantic analysis, that are commonly used in NLP. In recent times, there have been some work on spectral analysis of linguistic networks as well. Belkin and Goldsmith (2002) applied spectral analysis to understand the struture of morpho-syntactic networks of English words. The current work, on the other hand, is along the lines of Mukherjee et al. (2009), where the aim is to understand not only the principles of organization, but also the global topology of the network through the study of the spectrum. The most important contribution here, however, lies in the comparison of the topology of the syntactic and semantic DSNs, which, to the best of our knowledge, has not been explored previously.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 245–248,
Suntec, Singapore, 4 August 2009.

2 Network Construction

The syntactic and semantic DSNs are constructed from a raw text corpus. This work is restricted to the study of English DSNs only[1].

Syntactic DSN: We define our syntactic network in a similar way as previous works in unsupervised parts-of-speech induction (cf. (Schütze, 1995; Biemann, 2006)): The most frequent 200 words in the corpus (July 2008 dump of English Wikipedia) are used as features in a word window of ± 2 around the target words. Thus, each target word is described by an 800-dimensional feature vector, containing the number of times we observe one of the most frequent 200 words in the respective positions relative to the target word. In our experiments, we collect data for the most frequent 1000 and 5000 target words, arguing that all syntactic classes should be represented in those. A similarity measure between target words is defined by the cosine between the feature vectors. The syntactic graph is formed by inserting the target words as nodes and connecting nodes with edge weights equal to their cosine similarity if this similarity exceeds a threshold $t = 0.66$.

Semantic DSN: The construction of this network is inspired by (Lin, 1998). Specifically, we parsed a dump of English Wikipedia (July 2008) with the XLE parser (Riezler et al., 2002) and extracted the following dependency relations for nouns: Verb-Subject, Verb-Object, Noun-coordination, NN-compound, Adj-Mod. These lexicalized relations act as features for the nouns. Verbs are recorded together with their subcategorization frame, i.e. the same verb lemmas in different subcat frames would be treated as if they were different verbs. We compute log-likelihood significance between features and target nouns (as in (Dunning, 1993)) and keep only the most significant 200 features per target word. Each feature f gets a feature weight that is inversely proportional to the logarithm of the number of target words it applies on. The similarity of two target nouns is then computed as the sum of the feature weights they share. For our analysis, we restrict the graph to the most frequent 5000 target common nouns and keep only the 200 highest weighted edges per target noun. Note that the degree of a node can still be larger than 200 if this node is contained in many 200 highest weighted edges of other target nouns.

[1]As shown in (Nath et al., 2008), the basic structure of these networks are insensitive to minor variations in the parameters (e.g., thresholds and number of words) and the choice of distance metric.

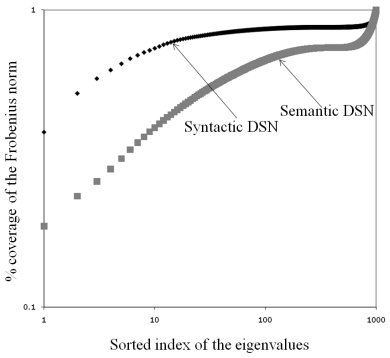


Figure 1: The spectrum of the syntactic and semantic DSNs of 1000 nodes.

3 Spectrum of DSNs

Spectral analysis refers to the systematic study of the eigenvalues and eigenvectors of a network. Although here we study the spectrum of the adjacency matrix of the weighted networks, it is also quite common to study the spectrum of the Laplacian of the adjacency matrix (see for example, Belkin and Goldsmith (2002)). Fig. 1 compares the spectrum of the syntactic and semantic DSNs with 1000 nodes, which has been computed as follows. First, the 1000 eigenvalues of the adjacency matrix are sorted in descending order. Then we compute the spectral coverage till the ith eigenvalue by adding the squares of the first i eigenvalues and normalizing it by the sum of the squares of all the eigenvalues - a quantity also known as the Frobenius norm of the matrix.

We observe that for the semantic DSN the first 10 eigenvalues cover only 40% of the spectrum and the first 500 together make up 75% of the spectrum. On the other hand, for the syntactic DSN, the first 10 eigenvalues cover 75% of the spectrum while the first 20 covers 80%. In other words, the structure of the syntactic DSN is governed by a few (order of 10) significant principles, whereas that of the semantic DSN is controlled by a large number of equally insignificant factors.

The aforementioned observation has the following alternative, but equivalent interpretations: (a) the syntactic DSN can be clustered in lower dimensions (e.g., 10 or 20) because, most of the rows in the matrix can be approximately expressed as a linear combination of the top 10 to 20

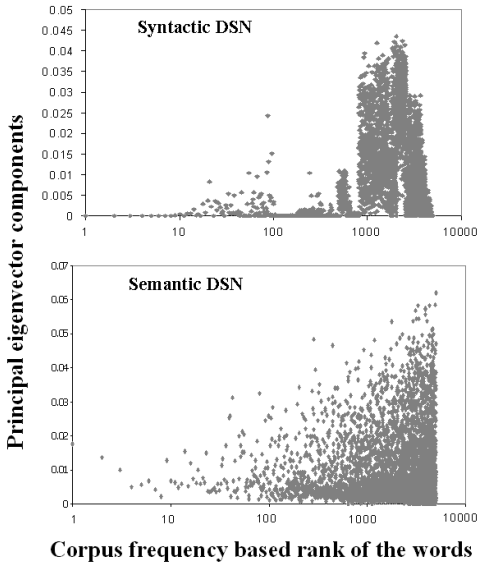


Figure 2: Plot of corpus frequency based rank vs. eigenvector centrality of the words in the DSNs of 5000 nodes.

eigenvectors. Furthermore, the graceful decay of the eigenvalues of the syntactic DSN implies the existence of a hierarchical community structure, which has been independently verified by Nath et al. (2008) through analysis of the degree distribution of such networks; and (b) a random walk conducted on the semantic DSN will have a high tendency to drift away very soon from the semantic class of the starting node, whereas in the syntactic DSN, the random walk is expected to stay within the same syntactic class for a long time. Therefore, it is reasonable to advocate that characterization and processing of syntatic classes is far less confusing than that of the semantic classes – a fact that requires no emphasis.

4 Eigenvector Analysis

The first eigenvalue tells us to what extent the rows of the adjacency matrix are correlated and therefore, the corresponding eigenvector is not a dimension pointing to any classificatory basis of the words. However, as we shall see shortly, the other eigenvectors corresponding to the significantly high eigenvalues are important classificatory dimensions.

Fig 2 shows the plot of the first eigenvector component (aka *eigenvector centrality*) of a word versus its rank based on the corpus frequency. We observe that the very high frequency (i.e., low rank) nodes in both the networks have low eigenvector centrality, whereas the medium frequency nodes display a wide range of centrality values. However, the most striking difference between the networks is that while in the syntactic DSN the centrality values are approximately normally distributed for the medium frequency words, the least frequent words enjoy the highest centrality for the semantic DSN. Furthermore, we observe that the most central nodes in the semantic DSN correspond to semantically unambiguous words of similar nature (e.g., deterioration, abandonment, fragmentation, turmoil). This indicates the existence of several "tightly knit communities consisting of not so high frequency words" which pull in a significant fraction of the overall centrality. Since the high frequency words are usually polysemous, they on the other hand form a large, but non-cliqueish structure at the core of the network with a few connections to the tightly knit communities. This is known as the tightly knit community effect (TKC effect) that renders very low centrality values to the "truly" central nodes of the network (Lempel and Moran, 2000). The structure of the syntactic DSN, however, is not governed by the TKC effect to such an extreme extent. Hence, one can expect to easily identify the natural classes of the syntactic DSN, but not its semantic counterpart.

In fact, this observation is further corroborated by the higher eigenvectors. Fig. 3 shows the plot of the second eigenvector component versus the fourth one for the two DSNs consisting of 5000 words. It is observed that for the syntactic network, the words get neatly clustered into two sets comprised of words with the positive and negative second eigenvector components. The same plot for the semantic DSN shows that a large number of words have both the components close to zero and only a few words stand out on one side of the axes – those with positive second eigenvector component and those with negative fourth eigenvector component. In essence, none of these eigenvectors can neatly classify the words into two sets – a trend which is observed for all the higher eigenvectors (we conducted experiments for up to the twentieth eigenvector).

Study of the individual eignevectors further reveals that the nodes with either the extreme positive or the extreme negative components have strong linguistic correlates. For instance, in the syntactic DSN, the two ends of the second eigen-

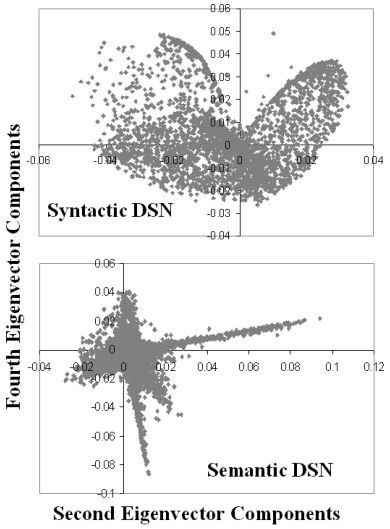


Figure 3: Plot of the second vs. fourth eigenvector components of the words in the DSNs.

vector correspond to nouns and adjectives; one of the ends of the fourth, fifth, sixth and the twelfth eigenvectors respectively correspond to location nouns, prepositions, first names and initials, and verbs. In the semantic DSN, one of the ends of the second, third, fourth and tenth eigenvectors respectively correspond to professions, abstract terms, food items and body parts. One would expect that the higher eigenvectors (say the 50^{th} one) would show no clear classificatory basis for the syntactic DSN, while for the semantic DSN those could be still associated with prominent linguistic correlates.

5 Conclusion and Future Work

Here, we presented some initial investigations into the nature of the syntactic and semantic DSNs through the method of spectral analysis, whereby we could observe that the global topology of the two networks are significantly different in terms of the organization of their natural classes. While the syntactic DSN seems to exhibit a hierarchical structure with a few strong natural classes and their mixtures, the semantic DSN is composed of several tightly knit small communities along with a large core consisting of very many smaller ill-defined and ambiguous sets of words. To visualize, one could draw an analogy of the syntactic and semantic DSNs respectively to "crystalline" and "amorphous" solids.

This work can be furthered in several directions, such as, (a) testing the robustness of the findings across languages, different network construction policies, and corpora of different sizes and from various domains; (b) clustering of the words on the basis of eigenvector components and using them in NLP applications such as unsupervised POS tagging and WSD; and (c) spectral analysis of WordNet and other manually constructed ontologies.

Acknowledgement

CB and AM are grateful to Microsoft Research India, respectively for hosting him while this research was conducted, and financial support.

References

M. Belkin and J. Goldsmith 2002. Using eigenvectors of the bigram graph to infer morpheme identity. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 4147, Association for Computational Linguistics.

Chris Biemann 2006. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the COLING/ACL-06 Student Research Workshop*.

Ted Dunning 1993. Accurate methods for the statistics of surprise and coincidence. In *Computational Linguistics* **19**, 1, pages 61–74

Z.S. Harris 1968. *Mathematical Structures of Language*. Wiley, New York.

R. Lempel and S. Moran 2000. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. In *Computer Networks*, **33**, pages 387-401

Dekang Lin 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING'98*.

Animesh Mukherjee, Monojit Choudhury and Ravi Kannan 2009. Discovering Global Patterns in Linguistic Networks through Spectral Analysis: A Case Study of the Consonant Inventories. In *The Proceedings of EACL 2009*, pages 585-593.

Joydeep Nath, Monojit Choudhury, Animesh Mukherjee, Christian Biemann and Niloy Ganguly 2008. Unsupervised parts-of-speech induction for Bengali. In *The Proceedings of LREC'08*, ELRA.

S. Riezler, T.H. King, R.M. Kaplan, R. Crouch, J.T. Maxwell, M. Johnson 2002. Parsing the Wall Street Journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 271-278.

Hinrich Schütze 1995. Distributional part-of-speech tagging. In *Proceedings of EACL*, pages 141-148.

Introduction of a new paraphrase generation tool based on Monte-Carlo sampling

Jonathan Chevelu[1,2] **Thomas Lavergne** **Yves Lepage**[1] **Thierry Moudenc**[2]
(1) GREYC, université de Caen Basse-Normandie
(2) Orange Labs; 2, avenue Pierre Marzin, 22307 Lannion
{jonathan.chevelu,thierry.moudenc}@orange-ftgroup.com,
thomas.lavergne@reveurs.org, yves.lepage@info.unicaen.fr

Abstract

We propose a new specifically designed method for paraphrase generation based on *Monte-Carlo* sampling and show how this algorithm is suitable for its task. Moreover, the basic algorithm presented here leaves a lot of opportunities for future improvement. In particular, our algorithm does not constraint the scoring function in opposite to *Viterbi* based decoders. It is now possible to use some global features in paraphrase scoring functions. This algorithm opens new outlooks for paraphrase generation and other natural language processing applications like statistical machine translation.

1 Introduction

A paraphrase generation system is a program which, given a source sentence, produces a different sentence with almost the same meaning.

Paraphrase generation is useful in applications to choose between different forms to keep the most appropriate one. For instance, automatic summary can be seen as a particular paraphrasing task (Barzilay and Lee, 2003) with the aim of selecting the shortest paraphrase.

Paraphrases can also be used to improve natural language processing (NLP) systems. (Callison-Burch et al., 2006) improved machine translations by augmenting the coverage of patterns that can be translated. Similarly, (Sekine, 2005) improved information retrieval based on pattern recognition by introducing paraphrase generation.

In order to produce paraphrases, a promising approach is to see the paraphrase generation problem as a translation problem, where the target language is the same as the source language (Quirk et al., 2004; Bannard and Callison-Burch, 2005).

A problem that has drawn less attention is the generation step which corresponds to the decoding step in SMT. Most paraphrase generation tools use some standard SMT decoding algorithms (Quirk et al., 2004) or some off-the-shelf decoding tools like MOSES (Koehn et al., 2007). The goal of a decoder is to find the best path in the lattice produced from a paraphrase table. This is basically achieved by using dynamic programming and especially the *Viterbi* algorithm associated with beam searching.

However decoding algorithms were designed for translation, not for paraphrase generation. Although left-to-right decoding is justified for translation, it may not be necessary for paraphrase generation. A paraphrase generation tool usually starts with a sentence which may be very similar to some potential solution. In other words, there is no need to "translate" all of the sentences. Moreover, decoding may not be suitable for non-contiguous transformation rules.

In addition, dynamic programming imposes an incremental scoring function to evaluate the quality of each hypothesis. For instance, it cannot capture some scattered syntactical dependencies. Improving on this major issue is a key point to improve paraphrase generation systems.

This paper first presents an alternative to decoding that is based on transformation rule application in section 2. In section 3 we propose a paraphrase generation method for this paradigm based on an algorithm used in two-player games. Section 4 briefly explain experimental context and its associated protocol for evaluation of the proposed system. We compare the proposed algorithm with a baseline system in section 5. Finally, in section 6, we point to future research tracks to improve paraphrase generation tools.

2 Statistical paraphrase generation using transformation rules

The paraphrase generation problem can be seen as an exploration problem. We seek the best paraphrase according to a scoring function in a space

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 249–252,
Suntec, Singapore, 4 August 2009.

to search by applying successive transformations. This space is composed of states connected by actions. An action is a transformation rule with a place where it applies in the sentence. States are a sentence with a set of possible actions. Applying an action in a given state consists in transforming the sentence of the state and removing all rules that are no more applicable. In our framework, each state, except the root, can be a final state. This is modelised by adding a stop rule as a particular action. We impose the constraint that any transformed part of the source sentence cannot be transformed anymore.

This paradigm is more approriate for paraphrase generation than the standard SMT approach in respect to several points: there is no need for left-to-right decoding because a transformation can be applied anywhere without order; there is no need to transform the whole of a sentence because each state is a final state; there is no need to keep the identity transformation for each phrase in the paraphrase table; the only domain knowledge needed is a generative model and a scoring function for final states; it is possible to mix different generative models because a statistical paraphrase table, an analogical solver and a paraphrase memory for instance; there is no constraint on the scoring function because it only scores final states.

Note that the branching factor with a paraphrase table can be around thousand actions per states which makes the generation problem a difficult computational problem. Hence we need an efficient generation algorithm.

3 Monte-Carlo based Paraphrase Generation

UCT (Kocsis and Szepesvári, 2006) (*Upper Confidence bound applied to Tree*) is a *Monte-Carlo* planning algorithm that have some interesting properties: it grows the search tree non-uniformly and favours the most promising sequences, without pruning branch; it can deal with high branching factor; it is an any-time algorithm and returns best solution found so far when interrupted; it does not require expert domain knowledge to evaluate states. These properties make it ideally suited for games with high branching factor and for which there is no strong evaluation function.

For the same reasons, this algorithm sounds interesting for paraphrase generation. In particular, it does not put constraint on the scoring function. We propose a variation of the UCT algorithm for paraphrase generation named MCPG for *Monte-Carlo based Paraphrase Generation*.

The main part of the algorithm is the sampling step. An episode of this step is a sequence of states and actions, $s_1, a_1, s_2, a_2, \ldots, s_T$, from the root state to a final state. During an episode construction, there are two ways to select the action a_i to perfom from a state s_i.

If the current state was already explored in a previous episode, the action is selected according to a compromise between exploration and exploitation. This compromise is computed using the UCB-Tunned formula (Auer et al., 2001) associated with the RAVE heuristic (Gelly and Silver, 2007). If the current state is explored for the first time, its score is estimated using *Monte-Carlo* sampling. In other word, to complete the episode, the actions $a_i, a_{i+1}, \ldots, a_{T-1}, a_T$ are selected randomly until a stop rule is drawn.

At the end of each episode, a reward is computed for the final state s_T using a scoring function and the value of each (state, action) pair of the episode is updated. Then, the algorithm computes an other episode with the new values.

Periodically, the sampling step is stopped and the best action at the root state is selected. This action is then definitely applied and a sampling is restarted from the new root state. The action sequence is built incrementally and selected after being enough sampled. For our experiments, we have chosen to stop sampling regularly after a fixed amount η of episodes.

Our main adaptation of the original algorithm is in the (state, action) value updating procedure. Since the goal of the algorithm is to maximise a scoring function, we use the maximum reachable score from a state as value instead of the score expectation. This algorithm suits the paradigm proposed for paraphrase generation.

4 Experimental context

This section describes the experimental context and the methodology followed to evaluate our statistical paraphrase generation tool.

4.1 Data

For the experiment reported in section 5, we use one of the largest, multi-lingual, freely available aligned corpus, Europarl (Koehn, 2005). It consists of European parliament debates. We choose

French as the language for paraphrases and English as the pivot language. For this pair of languages, the corpus consists of $1,487,459$ French sentences aligned with $1,461,429$ English sentences. Note that the sentences in this corpus are long, with an average length of 30 words per French sentence and 27.1 for English. We randomly extracted 100 French sentences as a test corpus.

4.2 Language model and paraphrase table

Paraphrase generation tools based on SMT methods need a language model and a paraphrase table. Both are computed on a training corpus.

The language models we use are n-gram language models with back-off. We use SRILM (Stolcke, 2002) with its default parameters for this purpose. The length of the n-grams is five.

To build a paraphrase table, we use the construction method via a pivot language proposed in (Bannard and Callison-Burch, 2005).

Three heuristics are used to prune the paraphrase table. The first heuristic prunes any entry in the paraphrase table composed of tokens with a probability lower than a threshold ϵ. The second, called *pruning pivot heuristic*, consists in deleting all pivot clusters larger than a threshold τ. The last heuristic keeps only the κ most probable paraphrases for each source phrase in the final paraphrase table. For this study, we empirically fix $\epsilon = 10^{-5}$, $\tau = 200$ and $\kappa = 10$.

4.3 Evaluation Protocol

We developed a dedicated website to allow the human judges with some flexibility in workplaces and evaluation periods. We retain the principle of the two-step evaluation, common in the machine translation domain and already used for paraphrase evaluation (Bannard and Callison-Burch, 2005).

The question asked to the human evaluator for the syntactic task is: *Is the following sentence in good French?* The question asked to the human evaluator for the semantic task is: *Do the following two sentences express the same thing?*

In our experiments, each paraphrase was evaluated by two native French evaluators.

5 Comparison with a SMT decoder

In order to validate our algorithm for paraphrase generation, we compare it with an off-the-shelf SMT decoder.

We use the MOSES decoder (Koehn et al., 2007) as a baseline. The MOSES scoring function is set by four weighting factors $\alpha_\Phi, \alpha_{LM}, \alpha_D, \alpha_W$. Conventionally, these four weights are adjusted during a tuning step on a training corpus. The tuning step is inappropriate for paraphrase because there is no such tuning corpus available. We empirically set $\alpha_\Phi = 1$, $\alpha_{LM} = 1$, $\alpha_D = 10$ and $\alpha_W = 0$. Hence, the scoring function (or reward function for MCPG) is equivalent to:

$$R(f'|f, I) = p(f') \times \Phi(f|f', I)$$

where f and f' are the source and target sentences, I a segmentation in phrases of f, $p(f')$ the language model score and $\Phi(f|f', I) = \prod_{i \in I} p(f^i|f'^i)$ the paraphrase table score.

The MCPG algorithm needs two parameters. One is the number of episodes η done before selecting the best action at root state. The other is k, an *equivalence parameter* which balances the exploration/exploitation compromise (Auer et al., 2001). We empirically set $\eta = 1,000,000$ and $k = 1,000$.

For our algorithm, note that identity paraphrase probabilities are biased: for each phrase it is equal to the probability of the most probable paraphrase. Moreover, as the source sentence is the best meaning preserved "paraphrase", a sentence cannot have a better score. Hence, we use a slightly different scoring function:

$$R(f'|f, I) = \min \left(\frac{p(f')}{p(f)} \prod_{\substack{i \in I \\ f^i \neq f'^i}} \frac{p(f^i|f'^i)}{p(f^i|f^i)}, 1 \right)$$

Note that for this model, there is no need to know the identity transformations probability for unchanged part of the sentence.

Results are presented in Table 1. The Kappa statistics associated with the results are 0.84, 0.64 and 0.59 which are usually considered as a "perfect", "substantial" and "moderate" agreement.

Results are close to evaluations from the baseline system. The main differences are from Kappa statistics which are lower for the MOSES system evaluation. Judges changed between the two experiments. We may wonder whether an evaluation with only two judges is reliable. This points to the ambiguity of any paraphrase definition.

System	MOSES	MCPG
Well formed (Kappa)	$64\%(0.57)$	$63\%(0.84)$
Meaning preserved (Kappa)	$58\%(0.48)$	$55\%(0.64)$
Well formed and meaning preserved (Kappa)	$50\%(0.54)$	$49\%(0.59)$

Table 1: Results of paraphrases evaluation for 100 sentences in French using English as the pivot language. Comparison between the baseline system MOSES and our algorithm MCPG.

By doing this experiment, we have shown that our algorithm with a biased paraphrase table is state-of-the-art to generate paraphrases.

6 Conclusions and further research

In this paper, we have proposed a different paradigm and a new algorithm in NLP field adapted for statistical paraphrases generation. This method, based on large graph exploration by *Monte-Carlo* sampling, produces results comparable with state-of-the-art paraphrase generation tools based on SMT decoders.

The algorithm structure is flexible and generic enough to easily work with discontinous patterns. It is also possible to mix various transformation methods to increase paraphrase variability.

The rate of ill-formed paraphrase is high at 37%. The result analysis suggests an involvement of the non-preservation of the original meaning when a paraphrase is evaluated ill-formed. Although the mesure is not statistically significant because the test corpus is too small, the same trend is also observed in other experiments. Improving on the language model issue is a key point to improve paraphrase generation systems. Our algorithm can work with unconstraint scoring functions, in particular, there is no need for the scoring function to be incremental as for *Viterbi* based decoders. We are working to add, in the scoring function, a linguistic knowledge based analyzer to solve this problem.

Because MCPG is based on a different paradigm, its output scores cannot be directly compared to MOSES scores. In order to prove the optimisation qualities of MCPG versus state-of-the-art decoders, we are transforming our paraphrase generation tool into a translation tool.

References

P. Auer, N. Cesa-Bianchi, and C. Gentile. 2001. Adaptive and self-confident on-line learning algorithms. *Machine Learning*.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Annual Meeting of ACL*, pages 597–604, Morristown, NJ, USA. Association for Computational Linguistics.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL 2003: Main Proceedings*, pages 16–23.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *HLT-NAACL 2006: Main Proceedings*, pages 17–24, Morristown, NJ, USA. Association for Computational Linguistics.

Sylvain Gelly and David Silver. 2007. Combining online and offline knowledge in UCT. In *24th International Conference on Machine Learning (*ICML*'07)*, pages 273–280, June.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *17th European Conference on Machine Learning, (*ECML*'06)*, pages 282–293, September.

Philipp Koehn, Hieu Hoang, Alexandra Birch Mayne, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of ACL, Demonstration Session*, pages 177–180, June.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*.

Chris Quirk, Chris Brockett, and Bill Dolan. 2004. Monolingual machine translation for paraphrase generation. In Dekang Lin and Dekai Wu, editors, *the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 142–149., Barcelona, Spain, 25-26 July. Association for Computational Linguistics.

Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of International Workshop on Paraphrase (IWP2005)*.

Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*.

Prediction of Thematic Rank for Structured Semantic Role Labeling

Weiwei Sun and **Zhifang Sui** and **Meng Wang**
Institute of Computational Linguistics
Peking University
Key Laboratory of Computational Linguistics
Ministry of Education, China
weiwsun@gmail.com;{wm,szf}@pku.edu.cn

Abstract

In Semantic Role Labeling (SRL), it is reasonable to globally assign semantic roles due to strong dependencies among arguments. Some relations between arguments significantly characterize the structural information of argument structure. In this paper, we concentrate on thematic hierarchy that is a rank relation restricting syntactic realization of arguments. A log-linear model is proposed to accurately identify thematic rank between two arguments. To import structural information, we employ re-ranking technique to incorporate thematic rank relations into local semantic role classification results. Experimental results show that automatic prediction of thematic hierarchy can help semantic role classification.

1 Introduction

In Semantic Role Labeling (SRL), it is evident that the arguments in one sentence are highly correlated. For example, a predicate will have no more than one *Agent* in most cases. It is reasonable to label one argument while taking into account other arguments. More structural information of all arguments should be encoded in SRL approaches.

This paper explores structural information of predicate-argument structure from the perspective of rank relations between arguments. Thematic hierarchy theory argues that there exists a language independent rank of possible semantic roles, which establishes priority among arguments with respect to their syntactic realization (Levin and Hovav, 2005). This construct has been widely implicated in linguistic phenomena, such as in the subject selection rule of Fillmore's Case Grammar (1968): "If there is an A [=Agent], it becomes the subject; otherwise, if there is an I [=Instrument], it becomes the subject; otherwise, the subject is the O [=Object, i.e., Patient/Theme]". This rule implicitly establishes precedence relations among semantic roles mentioned and can be simplified to:

$$Agent \succ Instrument \succ Patient/Theme$$

Emerging from a range of more basic semantic properties of the ranked semantic roles, thematic hierarchies can help to construct mapping from semantics to syntax. It is therefore an appealing option for argument structure analysis. For example, if the the rank of argument a_i is shown higher than a_j, then the assignment [a_i=*Patient*, a_j=*Agent*] is illegal, since the role *Agent* is the highest role.

We test the hypothesis that thematic rank between arguments can be accurately detected by using syntax clues. In this paper, the concept "thematic rank" between two arguments a_i and a_j means the relationship that a_i is prior to a_j or a_j is prior to a_i. Assigning different labels to different relations between a_i and a_j, we formulate prediction of thematic rank between two arguments as a multi-class classification task. A log-linear model is put forward for classification. Experiments on CoNLL-2005 data show that this approach can get an good performance, achieving 96.42% accuracy on gold parsing data and 95.14% accuracy on Charniak automatic parsing data.

Most existing SRL systems divide this task into two subtasks: Argument Identification (AI) and Semantic Role Classification (SRC). To add structural information to a local SRL approach, we incorporate thematic hierarchy relations into local classification results using re-ranking technique in the SRC stage. Two re-ranking approaches, 1) hard constraint re-ranking and 2) soft constraint re-ranking, are proposed to filter out unlike global semantic role assignment. Experiments on CoNLL-2005 data indicate that our method can yield significant improvement over a state-of-the-art SRC baseline, achieving 0.93% and 1.32%

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 253–256,
Suntec, Singapore, 4 August 2009.

absolute accuracy improvements on hand-crafted and automatic parsing data.

2 Prediction of Thematic Rank

2.1 Ranking Arguments in PropBank

There are two main problems in modeling thematic hierarchy for SRL on PropBank. On the one hand, there is no consistent meaning of the core roles (i.e. Arg0-5/ArgA). On the other hand, there is no consensus over hierarchies of the roles in the thematic hierarchy. For example, the *Patient* occupies the second highest hierarchy in some linguistic theories but the lowest in some other theories (Levin and Hovav, 2005).

In this paper, the proto-role theory (Dowty, 1991) is taken into account to rank PropBank arguments, partially resolving the two problems above. There are three key points in our solution. First, the rank of Arg0 is the highest. The *Agent* is almost without exception the highest role in proposed hierarchies. Though PropBank defines semantic roles on a verb by verb basis, for a particular verb, Arg0 is generally the argument exhibiting features of a prototypical *Agent* while Arg1 is a prototypical *Patient* or *Theme* (Palmer et al., 2005). As being the proto-Agent, the rank of Arg0 is higher than other numbered arguments. Second, the rank of the Arg1 is second highest or lowest. Both hierarchy of Arg1 are tested and discussed in section 4. Third, we do not rank other arguments.

Two sets of roles closely correspond to numbered arguments: 1) referenced arguments and 2) continuation arguments. To adapt the relation to help these two kinds of arguments, the equivalence relation is divided into several sub-categories. In summary, relations of two arguments a_i and a_j in this paper include: 1) $a_i \succ a_j$: a_i is higher than a_j, 2) $a_i \prec a_j$: a_i is lower than a_j, 3) $a_i ARa_j$: a_j is the referenced argument of a_i, 4) $a_i RAa_j$: a_i is the referenced argument of a_j, 5) $a_i ACa_j$: a_j is the continuation argument of a_i, 6) $a_i CAa_j$: a_i is the continuation argument of a_j, 7) $a_i = a_j$: a_i and a_j are labeled as the same role label, and 8) $a_i \sim a_j$: a_i and a_j are labeled as the Arg2-5, but not in the same type.

2.2 Prediction Method

Assigning different labels to possible rank between two arguments a_i and a_j, such as labeling $a_i \succ a_j$ as "$\succ$", identification of thematic rank can be formulated as a classification problem. Denote the set of relations $\mathcal{R}$. Formally, given a score function $S_{TH} : \mathcal{A} \times \mathcal{A} \times \mathcal{R} \mapsto \mathbb{R}$, the relation r is recognized in argmax flavor:

$$\hat{r} = r^*(a_i, a_j) = \arg\max_{r \in \mathcal{R}} S_{TH}(a_i, a_j, r)$$

A probability function is chosen as the score function and the log-linear model is used to estimate the probability:

$$S_{TH}(a_i, a_j, r) = \frac{\exp\{\psi(a_i, a_j, r) \cdot \mathbf{w}\}}{\sum_{r \in \mathcal{R}} \exp\{\psi(a_i, a_j, r) \cdot \mathbf{w}\}}$$

where ψ is the feature map and $\mathbf{w}$ is the parameter vector to learn. Note that the model predicts the rank of a_i and a_j through calculating $S_{TH}(a_i, a_j, r)$ rather than $S_{TH}(a_j, a_i, r)$, where a_i precedes a_j. In other words, the position information is implicitly encoded in the model rather than explicitly as a feature.

lemma, *POS Tag*, *voice*, and *SCF* of predicate
categories, *position* of two arguments; *rewrite rules* expanding subroots of two arguments
content and *POS tags* of the boundary words and head words
category path from the predicate to candidate arguments
single character category path from the predicate to candidate arguments
conjunction of categories, position, head words, POS of head words *category* and *single character category path* from the first argument to the second argument

Table 1: Features for thematic rank identification.

The system extracts a number of features to represent various aspects of the syntactic structure of a pair of arguments. All features are listed in Table 1. The *Path* features are designed as a sequential collection of phrase tags by (Gildea and Jurafsky, 2002). We also use *Single Character Category Path*, in which each phrase tag is clustered to a category defined by its first character (Pradhan et al., 2005). To characterize the relation between two constituents, we combine features of the two individual arguments as new features (i.e. conjunction features). For example, if the category of the first argument is *NP* and the category of the second is *S*, then the *conjunction of category* feature is *NP-S*.

3 Re-ranking Models for SRC

Toutanova et al. (2008) empirically showed that global information is important for SRL and that

structured solutions outperform local semantic role classifiers. Punyakanok et al. (2008) raised an inference procedure with integer linear programming model, which also showed promising results.

Identifying relations among arguments can provide structural information for SRL. Take the sentence "[$_{Arg0}$ She] [$_{V}$ addressed] [$_{Arg1}$ her husband] [$_{ArgM-MNR}$ with her favorite nickname]." for example, if the thematic rank of *she* and *her husband* is predicted as that *she* is higher than *her husband*, then *her husband* should not be assigned the highest role.

To incorporate the relation information to local classification results, we employ re-ranking approach. Assuming that the local semantic classifier can produce a list of labeling results, our system then attempts to pick one from this list according to the predicted ranks. Two different polices are implemented: 1) hard constraint re-ranking, and 2) soft constraint re-ranking.

Hard Constraint Re-ranking The one picked up must be strictly in accordance with the ranks. If the rank prediction result shows the rank of argument a_i is higher than a_j, then role assignments such as [a_i=*Patient* and a_j=*Agent*] will be eliminated. Formally, the score function of a global semantic role assignment is:

$$S(\mathbf{a},\mathbf{s}) = \prod_i S_l(a_i, s_i) \prod_{i,j,i<j} I(r^*(a_i,a_j), r(s_i,s_j))$$

where the function S_l locally scores an argument; $r^* : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{R}$ is to predict hierarchy of two arguments; $r : \mathcal{S} \times \mathcal{S} \mapsto \mathcal{R}$ is to point out the thematic hierarchy of two semantic roles. For example, $r(Agent, Patient) = " \succ "$. $I : \mathcal{R} \times \mathcal{R} \mapsto \{0, 1\}$ is identity function.

In some cases, there is no role assignment satisfies all predicted relations because of prediction mistakes. For example, if the hierarchy detection result of $\mathbf{a} = (a_1, a_2, a_3)$ is ($r^*(a_1, a_2) = \succ$, $r^*(a_2, a_3) = \succ$, $r^*(a_1, a_3) = \prec$), there will be no legal role assignment. In these cases, our system returns local SRL results.

Soft Constraint Re-ranking In this approach, the predicted confidence score of relations is added as factor items to the score function of the semantic role assignment. Formally, the score function in soft constraint re-ranking is:

$$S(\mathbf{a},\mathbf{s}) = \prod_i S_l(a_i, s_i) \prod_{i,j,i<j} S_{TH}(a_i, a_j, r(s_i,s_j))$$

4 Experiments

4.1 Experimental Settings

We evaluated our system using the CoNLL-2005 shared task data. Hierarchy labels for experimental corpora are automatically set according to the definition of relation labels described in section 2.1. Charniak parser (Charniak, 2000) is used for POS tagging and full parsing. UIUC Semantic Role Labeler [1] is a state-of-the-art SRL system. Its argument classification module is used as a strong local semantic role classifier. This module is retrained in our SRC experiments, using parameters described in (Koomen et al., 2005). Experiments of SRC in this paper are all based on good argument boundaries which can filter out the noise raised by argument identification stage.

4.2 Which Hierarchy Is Better?

	Detection	SRL (S)	SRL (G)
Baseline	–	94.77%	–
A	94.65%	95.44%	96.89%
A & P↑	95.62%	95.07%	96.39%
A & P↓	94.09%	95.13%	97.22%

Table 2: Accuracy on different hierarchies

Table 2 summarizes the performance of thematic rank prediction and SRC on different thematic hierarchies. All experiments are tested on development corpus. The first row shows the performance of the local sematic role classifier. The second to the forth rows show the performance based on three ranking approach. *A* means that the rank of *Agent* is the highest; *P*↑ means that the rank of *Patient* is the second highest; *P*↓ means that the rank of the *Patient* is the lowest. Column *SRL(S)* shows SRC performance based on soft constraint re-ranking approach, and column *SRL(G)* shows SRC performance based on gold hierarchies. The data shows that the third thematic hierarchy fits SRL best, but is harder to learn. Compared with P↑, P↓ is more suitable for SRL. In the following SRC experiments, we use the first hierarchy because it is most helpful when predicted relations are used.

4.3 Results And Improvement Analysis

Table 3 summarizes the precision, recall, and F-measure of this task. The second column is frequency of relations in the test data, which can be

[1] http://l2r.cs.uiuc.edu/∼cogcomp/srl-demo.php

seen as a simple baseline. Moreover, another natural baseline system can predict hierarchies according to the roles classified by local classifier. For example, if the a_i is labeled as Arg0 and a_j is labeled as Arg2, then the relation is predicted as $\succ$. The third column *BL* shows the F-measure of this baseline. It is clear that our approach significantly outperforms the two baselines.

Rel	Freq.	BL	P(%)	R(%)	F
$\succ$	57.40	94.79	97.13	98.33	97.73
$\prec$	9.70	51.23	98.52	97.24	97.88
$\sim$	23.05	13.41	94.49	93.59	94.04
=	0.33	19.57	93.75	71.43	81.08
AR	5.55	95.43	99.15	99.72	99.44
AC	3.85	78.40	87.77	82.04	84.81
CA	0.16	30.77	83.33	50.00	62.50
All	–	75.75	96.42		

Table 3: Thematic rank prediction performance

Table 4 summarizes overall accuracy of SRC. Baseline performance is the overall accuracy of the local classifier. We can see that our re-ranking methods can yield significant improvemnts over the baseline.

	Gold	Charniak
Baseline	95.14%	94.12%
Hard	95.71%	94.74%
Soft	96.07%	95.44%

Table 4: Overall SRC accuracy.

Hierarchy prediction and re-ranking can be viewed as modification for local classification results with structural information. Take the sentence "[Some 'circuit breakers' installed after the October 1987] crash failed [their first test]." for example, where phrases "*Some ... 1987*" and "*their ... test*" are two arguments. The table below shows the local classification result (column *Score(L)*) and the rank prediction result (column *Score(H)*). The baseline system falsely assigns roles as *Arg0+Arg1*, the rank relation of which is $\succ$. Taking into account rank prediction result that relation $\sim$ gets a extremely high probability, our system returns *Arg1+Arg2* as SRL result.

Assignment	Score(L)	Score(H)
Arg0+Arg1	$78.97\% \times 82.30\%$	$\succ$:0.02%
Arg1+Arg2	$14.25\% \times 11.93\%$	$\sim$:99.98%

5 Conclusion and Future Work

Inspired by thematic hierarchy theory, this paper concentrates on thematic hierarchy relation which characterize the structural information for SRL. The prediction of thematic rank is formulated as a classification problem and a log-linear model is proposed to solve this problem. To improve SRC, we employ re-ranking technique to incorporate thematic rank information into the local semantic role classifier. Experimental results show that our methods can construct high-performance thematic rank detector and that identification of arguments' relations can significantly improve SRC.

Acknowledgments

This work is supported by NSFC Project 60873156, 863 High Technology Project of China 2006AA01Z144 and the project of Toshiba (China) Co., Ltd. R&D Center.

References

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of NAACL-00*.

David R. Dowty. 1991. Thematic proto-roles and argument selection. *Language*, 67:547–619.

Charles Fillmore. 1968. The case for case. In Emmon Bach and Richard Harms, editors, *Universals in Linguistic Theory*, pages 1–90. Holt, Rinehart and Winston, New York, New York.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.

Peter Koomen, Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2005. Generalized inference with multiple semantic role labeling systems. In *Proceedings of the CoNLL-2005*, pages 181–184, June.

Beth Levin and Malka Rappaport Hovav. 2005. *Argument Realization*. Research Surveys in Linguistics. Cambridge University Press, New York.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. In *Machine Learning*.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Comput. Linguist.*

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Comput. Linguist.*

Transfer Learning, Feature Selection and Word Sense Disambguation

Paramveer S. Dhillon and Lyle H. Ungar
Computer and Information Science
University of Pennsylvania, Philadelphia, PA, U.S.A
{pasingh,ungar}@seas.upenn.edu

Abstract

We propose a novel approach for improving Feature Selection for Word Sense Disambiguation by incorporating a feature relevance prior for each word indicating which features are more likely to be selected. We use transfer of knowledge from similar words to learn this prior over the features, which permits us to learn higher accuracy models, particularly for the rarer word senses. Results on the ONTONOTES verb data show significant improvement over the baseline feature selection algorithm and results that are comparable to or better than other state-of-the-art methods.

1 Introduction

The task of WSD has been mostly studied in a supervised learning setting e.g. (Florian and Yarowsky, 2002) and feature selection has always been an important component of high accuracy word sense disambiguation, as one often has thousands of features but only hundreds of observations of the words (Florian and Yarowsky, 2002).

The main problem that arises with supervised WSD techniques, including ones that do feature selection, is the paucity of labeled data. For example, the training set of SENSEVAL-2 English lexical sample task has only 10 labeled examples per sense (Florian and Yarowsky, 2002), which makes it difficult to build high accuracy models using only supervised learning techniques. It is thus an attractive alternative to use transfer learning (Ando and Zhang, 2005), which improves performance by generalizing from solutions to "similar" learning problems. (Ando, 2006) (abbreviated as Ando[CoNLL'06]) have successfully applied the ASO (Alternating Structure Optimization) technique proposed by (Ando and Zhang, 2005), in its transfer learning configuration, to the problem of WSD by doing joint empirical risk minimization of a set of related problems (words in this case). In this paper, we show how a novel form of transfer learning that learns a feature relevance prior from similar word senses, aids in the process of feature selection and hence benefits the task of WSD.

Feature selection algorithms usually put a uniform prior over the features. I.e., they consider each feature to have the same probability of being selected. In this paper we relax this overly simplistic assumption by transferring a prior for feature relevance of a given word sense from "similar" word senses. Learning this prior for feature relevance of a test word sense makes those features that have been selected in the models of other "similar" word senses become more likely to be selected.

We learn the feature relevance prior only from distributionally similar word *senses*, rather than "all" senses of each word, as it is difficult to find words which are similar in "all" the senses. We can, however, often find words which have one or a few similar senses. For example, one sense of "fire" (as in "fire someone") should share features with one sense of "dismiss" (as in "dismiss someone"), but other senses of "fire" (as in "fire the gun") do not. Similarly, other meanings of "dismiss" (as in "dismiss an idea") should not share features with "fire".

As just mentioned, knowledge can only be fruitfully transfered between the shared senses of different words, even though the models being learned are for disambiguating different senses of a single word. To address this problem, we cluster similar word senses of different words, and then use the models learned for all but one of the word senses in the cluster (called the "training word senses") to put a feature relevance prior on which features will be more predictive for the held out test word sense. We hold out each word sense in the cluster once and learn a prior from the remaining word senses in that cluster. For example, we can use the models for discriminating the senses of the words "kill" and the senses of "capture", to

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 257–260,
Suntec, Singapore, 4 August 2009.

put a prior on what features should be included in a model to disambiguate corresponding senses of the distributionally similar word "arrest".

The remainder of the paper is organized as follows. In Section 2 we describe our "baseline" information theoretic feature selection method, and extend it to our "TRANSFEAT" method. Section 3 contains experimental results comparing TRANSFEAT with the baseline and Ando[CoNLL'06] on ONTONOTES data. We conclude with a brief summary in Section 4.

2 Feature Selection for WSD

We use an information theoretic approach to feature selection based on the Minimum Description Length (MDL) (Rissanen, 1999) principle, which makes it easy to incorporate information about feature relevance priors. These information theoretic models have a 'dual' Bayesian interpretation, which provides a clean setting for feature selection.

2.1 Information Theoretic Feature Selection

The state-of-the-art feature selection methods in WSD use either an ℓ_0 or an ℓ_1 penalty on the coefficients. ℓ_1 penalty methods such as Lasso, being convex, can be solved by optimization and give guaranteed optimal solutions. On the other hand, ℓ_0 penalty methods, like stepwise feature selection, give approximate solutions but produce models that are much sparser than the models given by ℓ_1 methods, which is quite crucial in WSD (Florian and Yarowsky, 2002). ℓ_0 models are also more amenable to theoretical analysis for setting thresholds, and hence for incorporating priors.

Penalized likelihood methods which are widely used for feature selection minimize a score:

$$Score = -2log(likelihood) + Fq \qquad (1)$$

where *F* is a function designed to penalize model complexity, and *q* represents the number of features currently included in the model at a given point. The first term in the above equation represents a measure of the in-sample error given the model, while the second term is a model complexity penalty.

As is obvious from Eq. 1, the description length of the MDL (Minimum Description Length) message is composed of two parts: S_E, the number of bits for encoding the residual errors given the models and S_M, the number of bits for encoding the model. Hence the description length can be written as: $S = S_E + S_M$. Now, when we evaluate a feature for possible addition to our model, we want to **maximize** the reduction of "description length" incurred by adding this feature to the model. This change in description length is: $\Delta S = \Delta S_E - \Delta S_M$; where $\Delta S_E \geq 0$ is the number of bits saved in describing residual error due to increase in the likelihood of the data given the new feature and $\Delta S_M > 0$ is the extra bits used for coding this new feature.

In our baseline feature selection model, we use the following coding schemes:

Coding Scheme for S_E :

The term S_E represents the cost of coding the residual errors given the models and can be written as:

$$S_E = -\log(P(y|w,x))$$

ΔS_E represents the increase in likelihood (in bits) of the data by adding this new feature to the model. We assume a Gaussian model, giving:

$$P(y|w,x) \sim \exp\left(-\left(\frac{\sum_{i=1}^{n}(y_i - w \cdot x_i)^2}{2\sigma^2}\right)\right)$$

where *y* is the response (word senses in our case), *x's* are the features, *w's* are the regression weights and σ^2 is the variance of the Gaussian noise.

Coding Scheme for ΔS_M : For describing S_M, the number of bits for encoding the model, we need the bits to code the index of the feature (i.e., which feature from amongst the total m candidate features) and the bits to code the coefficient of this feature.

The total cost can be represented as:

$$S_M = l_f + l_\theta$$

where l_f is the cost to code the index of the feature and l_θ is the number of bits required to code the coefficient of the selected feature.

In our baseline feature selection algorithm, we code l_f by using $log(m)$ bits (where m is the total number of candidate features), which is equivalent to the standard RIC (or the Bonferroni penalty) (Foster and George, 1994) commonly used in information theory. The above coding scheme[1] corresponds to putting a uniform prior over all the features; I.e., each feature is equally likely to get selected.

For coding the coefficients of the selected feature we use 2 bits, which is quite similar to the AIC

[1]There is a duality between Information Theory and Bayesian terminology: If there is $\frac{1}{k}$ probability of a fact being true, then we need $-log(\frac{1}{k}) = log(k)$ bits to code it.

(Akaike Information Criterion) (Rissanen, 1999). Our final equation for S_M is therefore:

$$S_M = log(m) + 2 \quad (2)$$

2.2 Extension to TRANSFEAT

We now extend the baseline feature selection algorithm to include the feature relevance prior. We define a binary random variable $f_i \in \{0,1\}$ that denotes the event of the i^{th} feature being in or not being in the model for the test word sense. We can parameterize the distribution as $p(f_i = 1|\theta_i) = \theta_i$. I.e., we have a Bernoulli Distribution over the features.

Given the data for the i^{th} feature for all the training word senses, we can write: $\mathcal{D}_i = \{f_{i1}, ..., f_{iv}, ..., f_{it}\}$. We then construct the likelihood functions from the data (under the i.i.d assumption) as:

$$p(\mathcal{D}_{f_i}|\theta_i) = \prod_{v=1}^{t} p(f_{iv}|\theta_i) = \prod_{v=1}^{t} \theta^{f_{iv}}(1-\theta_i)^{1-f_{iv}}$$

The posteriors can be calculated by putting a prior over the parameters θ_i and using Bayes rule as follows:

$$p(\theta_i|\mathcal{D}_{f_i}) = p(\mathcal{D}_{f_i}|\theta_i) \times p(\theta_i|a, b)$$

where a and b are the hyperparameters of the Beta Prior (conjugate of Bernoulli). The predictive distribution of θ_i is:

$$\begin{aligned} p(f_i = 1|\mathcal{D}_{f_i}) &= \int_0^1 \theta_i p(\theta_i|\mathcal{D}_{f_i}) d\theta_i = \mathbb{E}[\theta_i|\mathcal{D}_{f_i}] \\ &= \frac{k+a}{k+l+a+b} \end{aligned} \quad (3)$$

where k is the number of times that the i^{th} feature is selected and l is the complement of k, i.e. the number of times the i^{th} feature is not selected in the training data.

In light of above, the coding scheme, which incorporates the prior information about the predictive quality of the various features obtained from similar word senses, can be formulated as follows:

$$S_M = -\log\left(p(f_i = 1|\mathcal{D}_{f_i})\right) + 2$$

In the above equation, the first term represents the cost of coding the features, and the second term codes the coefficients. The negative signs appear due to the duality between Bayesian and Information-Theoretic representation, as explained earlier.

3 Experimental Results

In this section we present the experimental results of TRANSFEAT on ONTONOTES data.

3.1 Similarity Determination

To determine which verbs to transfer from, we cluster verb senses into groups based on the TF/IDF similarity of the vector of features selected for that verb sense in the baseline (non-transfer learning) model. We use only those features that are positively correlated with the given sense; they are the features most closely associated with the given sense. We cluster senses using a "foreground-background" clustering algorithm (Kandylas et al., 2007) rather than the more common k-means clustering because many word senses are not sufficiently similar to any other word sense to warrant putting into a cluster. Foreground-background clustering gives highly cohesive clusters of word senses (the "foreground") and puts all the remaining word senses in the "background". The parameters that it takes as input are the $\%$ of data points to put in "background" (i.e., what would be the singleton clusters) and a similarity threshold which impacts the number of "foreground" clusters. We experimented with putting 20% and 33% data points in background and adjusted the similarity threshold to give us $50 - 100$ "foreground" clusters. The results reported below have 20% background and $50 - 100$ "foreground" clusters.

3.2 Description of Data and Results

We performed our experiments on ONTONOTES data of 172 verbs (Hovy et al., 2006). The data consists of a rich set of linguistic features which have proven to be beneficial for WSD.

A sample feature vector for the word "add", given below, shows typical features.

```
word_added pos_vbd morph_normal
subj_use subjsyn_16993 dobj_money
dobjsyn_16993 pos+1+2+3_rp+to+cd
tp_account tp_accumulate tp_actual
```

The 172 verbs each had between 1,000 and 10,000 nonzero features. The number of senses varied from 2 (For example, "add") to 15 (For example, "turn").

We tested our transfer learning algorithm in three slightly varied settings to tease apart the contributions of different features to the overall performance. In our main setting, we cluster the word

senses based on the "semantic + syntactic" features. In Setting 2, we do clustering based only on "semantic" features (topic features) and in Setting 3 we cluster based on only "syntactic" (pos, dobj etc.) features.

Table 1: 10-fold CV (microaveraged) accuracies of various methods for various Transfer Learning settings. **Note:** These are true cross-validation accuracies; No parameters have been tuned on them.

Method	Setting 1	Setting 2	Setting 3
TRANSFEAT	85.75	85.11	85.37
Baseline Feat. Sel.	83.50	83.09	83.34
SVM (Poly. Kernel)	83.77	83.44	83.57
Ando[CoNLL'06]	85.94	85.00	85.51
Most Freq. Sense	76.59	77.14	77.24

We compare TRANSFEAT against Baseline Feature Selection, Ando[CoNLL'06], SVM (libSVM package) with a cross-validated polynomial kernel and a simple most frequent sense baseline. We tuned the "d" parameter of the polynomial kernel using a separate cross validation.

The results for the different settings are shown in Table 1 and are significantly better at the 5% significance level (Paired t-test) than the baseline feature selection algorithm and the SVM. It is comparable in accuracy to Ando[CoNLL'06]. Settings 2 and 3, in which we cluster based on only "semantic" or "syntactic" features, respectively, also gave significant (5% level in a Paired t-Test) improvement in accuracy over the baseline and SVM model. But these settings performed slightly worse than Setting 1, which suggests that it is a good idea to have clusters in which the word senses have "semantic" as well as "syntactic" distributional similarity.

Some examples will help to emphasize the point that we made earlier that transfer helps the most in cases in which the target word sense has much less data than the word senses from which knowledge is being transferred. "kill" had roughly 6 times more data than all other word senses in its cluster (i.e., "arrest", "capture", "strengthen", etc.) In this case, TRANSFEAT gave $3.19 - 8.67\%$ higher accuracies than competing methods[2] on these three words. Also, for the case of word "do," which had roughly 10 times more data than the other word senses in its cluster (E.g., "die" and "save"), TRANSFEAT gave $4.09 - 6.21\%$ higher accuracies than other methods. Transfer makes the biggest difference when the target words have much less data than the word senses they are generalizing from, but even in cases where the words have similar amounts of data we still get a $1.5 - 2.5\%$ increase in accuracy.

[2] TRANSFEAT does better than Ando[CoNLL'06] on these words even though on average over all 172 verbs, the difference is slender.

4 Summary

This paper presented a Transfer Learning formulation which learns a prior suggesting which features are most useful for disambiguating ambiguous words. Successful transfer requires finding similar word senses. We used "foreground/background" clustering to find cohesive clusters for various word senses in the ONTONOTES data, considering both "semantic" and "syntactic" similarity between the word senses. Learning priors on features was found to give significant accuracy boosts, with both syntactic and semantic features contributing to successful transfer. Both feature sets gave substantial benefits over the baseline methods that did not use any transfer and gave comparable accuracy to recent Transfer Learning methods like Ando[CoNLL'06]. The performance improvement of our Transfer Learning becomes even more pronounced when the word senses that we are generalizing from have more observations than the ones that are being learned.

References

R. Ando and T. Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6:1817–1853.

R. Ando. 2006. Applying alternating structure optimization to word sense disambiguation. In *(CoNLL)*.

R. Florian and D. Yarowsky. 2002. Modeling consensus: classifier combination for word sense disambiguation. In *EMNLP '02*, pages 25–32.

D. P. Foster and E. I. George. 1994. The risk inflation criterion for multiple regression. *The Annals of Statistics*, 22(4):1947–1975.

E. H. Hovy, M. P. Marcus, M. Palmer, L. A. Ramshaw, and R. M. Weischedel. 2006. Ontonotes: The 90% solution. In *HLT-NAACL*.

V. Kandylas, S. P. Upham, and L. H. Ungar. 2007. Finding cohesive clusters for analyzing knowledge communities. In *ICDM*, pages 203–212.

J. Rissanen. 1999. Hypothesis selection and testing by the mdl principle. *The Computer Journal*, 42:260–269.

From Extractive to Abstractive Meeting Summaries: Can It Be Done by Sentence Compression?

Fei Liu and Yang Liu
Computer Science Department
The University of Texas at Dallas
Richardson, TX 75080, USA
{feiliu, yangl}@hlt.utdallas.edu

Abstract

Most previous studies on meeting summarization have focused on extractive summarization. In this paper, we investigate if we can apply sentence compression to extractive summaries to generate abstractive summaries. We use different compression algorithms, including integer linear programming with an additional step of filler phrase detection, a noisy-channel approach using Markovization formulation of grammar rules, as well as human compressed sentences. Our experiments on the ICSI meeting corpus show that when compared to the abstractive summaries, using sentence compression on the extractive summaries improves their ROUGE scores; however, the best performance is still quite low, suggesting the need of language generation for abstractive summarization.

1 Introduction

Meeting summaries provide an efficient way for people to browse through the lengthy recordings. Most current research on meeting summarization has focused on extractive summarization, that is, it extracts important sentences (or dialogue acts) from speech transcripts, either manual transcripts or automatic speech recognition (ASR) output. Various approaches to extractive summarization have been evaluated recently. Popular unsupervised approaches are maximum marginal relevance (MMR), latent semantic analysis (LSA) (Murray et al., 2005a), and integer programming (Gillick et al., 2009). Supervised methods include hidden Markov model (HMM), maximum entropy, conditional random fields (CRF), and support vector machines (SVM) (Galley, 2006; Buist et al., 2005; Xie et al., 2008; Maskey and Hirschberg, 2006). (Hori et al., 2003) used a word based speech summarization approach that utilized dynamic programming to obtain a set of words to maximize a summarization score.

Most of these summarization approaches aim for selecting the most informative sentences, while less attempt has been made to generate abstractive summaries, or compress the extracted sentences and merge them into a concise summary. Simply concatenating extracted sentences may not comprise a good summary, especially for spoken documents, since speech transcripts often contain many disfluencies and are redundant. The following example shows two extractive summary sentences (they are from the same speaker), and part of the abstractive summary that is related to these two extractive summary sentences. This is an example from the ICSI meeting corpus (see Section 2.1 for more information on the data).

Extractive summary sentences:
Sent1: um we have to refine the tasks more and more which of course we haven't done at all so far in order to avoid this rephrasing
Sent2: and uh my suggestion is of course we we keep the wizard because i think she did a wonderful job

Corresponding abstractive summary:
the group decided to hire the wizard and continue with the refinement...

In this paper, our goal is to answer the question if we can perform sentence compression on an extractive summary to improve its readability and make it more like an abstractive summary. Compressing sentences could be a first step toward our ultimate goal of creating an abstract for spoken documents. Sentence compression has been widely studied in language processing. (Knight and Marcu, 2002; Cohn and Lapata, 2009) learned rewriting rules that indicate which words should be dropped in a given context. (Knight and Marcu, 2002; Turner and Charniak, 2005) applied the noisy-channel framework to predict the possibilities of translating a sentence to a shorter word sequence. (Galley and McKeown, 2007) extended the noisy-channel approach and proposed a head-driven Markovization formulation of synchronous context-free grammar (SCFG) deletion rules. Unlike these approaches that need a training corpus, (Clarke and Lapata, 2008) encoded the language model and a variety of linguistic constraints as linear inequalities, and employed the integer programming approach to find a subset of words that maximize an objective function.

Our focus in this paper is not on new compression algorithms, but rather on using compression to bridge the gap of extractive and abstractive summarization. We use different automatic compression algorithms. The first one is the integer programming (IP) framework, where we also introduce a filler phrase (FP) detection

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 261–264,
Suntec, Singapore, 4 August 2009.

module based on the Web resources. The second one uses the SCFG that considers the grammaticality of the compressed sentences. Finally, as a comparison, we also use human compression. All of these compressed sentences are compared to abstractive summaries. Our experiments using the ICSI meeting corpus show that compressing extractive summaries can improve human readability and the ROUGE scores against the reference abstractive summaries.

2 Sentence Compression of Extractive Summaries

2.1 Corpus

We used the ICSI meeting corpus (Janin et al., 2003), which contains naturally occurring meetings, each about an hour long. All the meetings have been transcribed and annotated with dialogue acts (DAs), topics, abstractive and extractive summaries (Shriberg et al., 2004; Murray et al., 2005b). In this study, we use the extractive and abstractive summaries of 6 meetings from this corpus. These 6 meetings were chosen because they have been used previously in other related studies, such as summarization and keyword extraction (Murray et al., 2005a). On average, an extractive summary contains 76 sentences[1] (1252 words), and an abstractive summary contains 5 sentences (111 words).

2.2 Compression Approaches

2.2.1 Human Compression

The data annotation was conducted via Amazon Mechanical Turk[2]. Human annotators were asked to generate condensed version for each of the DAs in the extractive summaries. The compression guideline is similar to (Clarke and Lapata, 2008). The annotators were asked to only remove words from the original sentence while preserving most of the important meanings, and make the compressed sentence as grammatical as possible. The annotators can leave the sentence uncompressed if they think no words need to be deleted; however, they were not allowed to delete the entire sentence. Since the meeting transcripts are not as readable as other text genres, we may need a better compression guideline for human annotators. Currently we let the annotators make their own judgment what is an appropriate compression for a spoken sentence.

We split each extractive meeting summary sequentially into groups of 10 sentences, and asked 6 to 10 online workers to compress each group. Then from these results, another human subject selected the best annotation for each sentence. We also asked this human judge to select the 4-best compressions. However, in this study, we only use the 1-best annotation result. We would like to do more analysis on the 4-best results in the future.

[1]The extractive units are DAs. We use DAs and sentences interchangeably in this paper when there is no ambiguity.

[2]http://www.mturk.com/mturk/welcome

2.2.2 Filler Phrase Detection

We define filler phrases (FPs) as the combination of two or more words, which could be discourse markers (e.g., I mean, you know), editing terms, as well as some terms that are commonly used by human but without critical meaning, such as, "for example", "of course", and "sort of". Removing these fillers barely causes any information loss. We propose to use web information to automatically generate a list of filler phrases and filter them out in compression.

For each extracted summary sentence of the 6 meetings, we use it as a query to Google and examine the top N returned snippets (N is 400 in our experiments). The snippets may not contain all the words in a sentence query, but often contain frequently occurring phrases. For example, "of course" can be found with high frequency in the snippets. We collect all the phrases that appear in both the extracted summary sentences and the snippets with a frequency higher than three. Then we calculate the inverse sentence frequency (ISF) for these phrases using the entire ICSI meeting corpus. The ISF score of a phrase i is:

$$isf_i = \frac{N}{N_i}$$

where N is the total number of sentences and N_i is the number of sentences containing this phrase. Phrases with low ISF scores mean that they appear in many occasions and are not domain- or topic-indicative. These are the filler phrases we want to remove to compress a sentence. The three phrases we found with the lowest ISF scores are "you know", "i mean" and "i think", consistent with our intuition.

We also noticed that not all the phrases with low ISF scores can be taken as FPs ("we are" would be a counter example). We therefore gave the ranked list of FPs (based on ISF values) to a human subject to select the proper ones. The human annotator crossed out the phrases that may not be removable for sentence compression, and also generated simple rules to shorten some phrases (such as turning "a little bit" into "a bit"). This resulted in 50 final FPs and about a hundred simplification rules. Examples of the final FPs are: 'you know', 'and I think', 'some of', 'I mean', 'so far', 'it seems like', 'more or less', 'of course', 'sort of', 'so forth', 'I guess', 'for example'. When using this list of FPs and rules for sentence compression, we also require that an FP candidate in the sentence is considered as a phrase in the returned snippets by the search engine, and its frequency in the snippets is higher than a pre-defined threshold.

2.2.3 Compression Using Integer Programming

We employ the integer programming (IP) approach in the same way as (Clarke and Lapata, 2008). Given an utterance $S = w_1, w_2, ..., w_n$, the IP approach forms a compression of this utterance only by dropping words and preserving the word sequence that maximizes an objective function, defined as the sum of the signifi-

cance scores of the consisting words and n-gram probabilities from a language model:

$$\max \lambda \cdot \sum_{i=1}^{n} y_i \cdot Sig(w_i) + (1-\lambda) \cdot \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} x_{ijk} \cdot P(w_k|w_i, w_j)$$

where y_i and x_{ijk} are two binary variables: $y_i = 1$ represents that word w_i is in the compressed sentence; $x_{ijk} = 1$ represents that the sequence w_i, w_j , w_k is in the compressed sentence. A trade-off parameter λ is used to balance the contribution from the significance scores for individual words and the language model scores. Because of space limitation, we omitted the special sentence beginning and ending symbols in the formula above. More details can be found in (Clarke and Lapata, 2008). We only used linear constraints defined on the variables, without any linguistic constraints.

We use the lp_solve toolkit.[3] The significance score for each word is its TF-IDF value (term frequency $\times$ inverse document frequency). We trained a language model using SRILM [4] on broadcast news data to generate the trigram probabilities. We empirically set λ as 0.7, which gives more weight to the word significance scores. This IP compression method is applied to the sentences after filler phrases (FPs) are filtered out. We refer to the output from this approach as "FP + IP".

2.2.4 Compression Using Lexicalized Markov Grammars

The last sentence compression method we use is the lexicalized Markov grammar-based approach (Galley and McKeown, 2007) with edit word detection (Charniak and Johnson, 2001). Two outputs were generated using this method with different compression rates (defined as the number of words preserved in the compression divided by the total number of words in the original sentence).[5] We name them "Markov (S1)" and "Markov (S2)" respectively.

3 Experiments

First we perform human evaluation for the compressed sentences. Again we use the Amazon Mechanical Turk for the subjective evaluation process. For each extractive summary sentence, we asked 10 human subjects to rate the compressed sentences from the three systems, as well as the human compression. This evaluation was conducted on three meetings, containing 244 sentences in total. Participants were asked to read the original sentence and assign scores to each of the compressed sentences for its informativeness and grammaticality respectively using a 1 to 5 scale. An overall score is calculated as the average of the informativeness and grammaticality scores. Results are shown in Table 1.

[3] http://www.geocities.com/lpsolve

[4] http://www.speech.sri.com/projects/srilm/

[5] Thanks to Michel Galley to help generate these output.

For a comparison, we also include the ROUGE-1 F-scores (Lin, 2004) of each system output against the human compressed sentences.

Approach	Info.	Gram.	Overall	R-1 F (%)
Human	4.35	4.38	4.37	-
Markov (S1)	3.64	3.79	3.72	88.76
Markov (S2)	2.89	2.76	2.83	62.99
FP + IP	3.70	3.95	3.82	85.83

Table 1: Human evaluation results. Also shown is the ROUGE-1 (unigram match) F-score of different systems compared to human compression.

We can see from the table that as expected, the human compression yields the best performance on both informativeness and grammaticality. 'FP + IP' and 'Markov (S1)' approaches also achieve satisfying performance under both evaluation metrics. The relatively low scores for 'Markov (S2)' output are partly due to its low compression rate (see Table 2 for the length information). As an example, we show below the compressed sentences from human and systems for the first sentence in the example in Sec 1.

Human: we have to refine the tasks in order to avoid rephrasing
Markov (S1): we have to refine the tasks more and more which we haven't done in order to avoid this rephrasing
Markov (S2): we have to refine the tasks which we haven't done order to avoid this rephrasing
FP + IP: we have to refine the tasks more and more which we haven't done to avoid this rephrasing

Since our goal is to answer the question if we can use sentence compression to generate abstractive summaries, we compare the compressed summaries, as well as the original extractive summaries, against the reference abstractive summaries. The ROUGE-1 results along with the word compression ratio for each compression approach are shown in Table 2. We can see that all of the compression algorithms yield better ROUGE score than the original extractive summaries. Take Markov (S2) as an example. The recall rate dropped only 8% (from the original 66% to 58%) when only 53% words in the extractive summaries are preserved. This demonstrates that it is possible for the current sentence compression systems to greatly condense the extractive summaries while preserving the desirable information, and thus yield summaries that are more like abstractive summaries. However, since the abstractive summaries are much shorter than the extractive summaries (even after compression), it is not surprising to see the low precision results as shown in Table 2. We also observe some different patterns between the ROUGE scores and the human evaluation results in Table 1. For example, Markov (S2) has the highest ROUGE result, but worse human evaluation score than other methods.

To evaluate the length impact and to further make

	All Sent.				Top Sent.		
Approach	Word ratio (%)	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)
Original extractive summary	100	7.58	66.06	12.99	29.98	34.29	31.83
Human compression	65.58	10.43	63.00	16.95	34.35	37.39	35.79
Markov (S1)	67.67	10.15	61.98	16.41	34.24	36.88	35.46
Markov (S2)	53.28	11.90	58.14	18.37	32.23	34.96	33.49
FP + IP	76.38	9.11	59.85	14.78	31.82	35.62	33.57

Table 2: Compression ratio of different systems and ROUGE-1 scores compared to human abstractive summaries.

the extractive summaries more like abstractive summaries, we conduct an oracle experiment: we compute the ROUGE score for each of the extractive summary sentences (the original sentence or the compressed sentence) against the abstract, and select the sentences with the highest scores until the number of selected words is about the same as that in the abstract.[6] The ROUGE results using these selected top sentences are shown in the right part of Table 2. There is some difference using all the sentences vs. the top sentences regarding the ranking of different compression algorithms (comparing the two blocks in Table 2).

From Table 2, we notice significant performance improvement when using the selected sentences to form a summary. These results indicate that, it may be possible to convert extractive summaries to abstractive summaries. On the other hand, this is an oracle result since we compare the extractive summaries to the abstract for sentence selection. In the real scenario, we will need other methods to rank sentences. Moreover, the current ROUGE score is not very high. This suggests that there is a limit using extractive summarization and sentence compression to form abstractive summaries, and that sophisticated language generation is still needed.

4 Conclusion

In this paper, we attempt to bridge the gap between extractive and abstractive summaries by performing sentence compression. Several compression approaches are employed, including an integer programming based framework, where we also introduced a filler phrase detection module, the lexicalized Markov grammar-based approach, as well as human compression. Results show that, while sentence compression provides a promising way of moving from extractive summaries toward abstracts, there is also a potential limit along this direction. This study uses human annotated extractive summaries. In our future work, we will evaluate using automatic extractive summaries. Furthermore, we will explore the possibility of merging compressed extractive sentences to generate more unified summaries.

References

A. Buist, W. Kraaij, and S. Raaijmakers. 2005. Automatic summarization of meeting data: A feasibility study. In *Proc. of CLIN*.

E. Charniak and M. Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proc. of NAACL*.

J. Clarke and M. Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

T. Cohn and M. Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*.

M. Galley and K. McKeown. 2007. Lexicalized markov grammars for sentence compression. In *Proc. of NAACL/HLT*.

M. Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proc. of EMNLP*.

D. Gillick, K. Riedhammer, B. Favre, and D. Hakkani-Tur. 2009. A global optimization framework for meeting summarization. In *Proc. of ICASSP*.

C. Hori, S. Furui, R. Malkin, H. Yu, and A. Waibel. 2003. A statistical approach to automatic speech summarization. *Journal on Applied Signal Processing*, 2003:128–139.

A. Janin, D. Baron, J. Edwards, D. Ellis, G. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003. The ICSI meeting corpus. In *Proc. of ICASSP*.

K. Knight and D. Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139:91–107.

C. Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. of ACL Workshop on Text Summarization Branches Out*.

S. Maskey and J. Hirschberg. 2006. Summarizing speech without text using hidden markov models. In *Proc. of HLT/NAACL*.

G. Murray, S. Renals, and J. Carletta. 2005a. Extractive summarization of meeting recordings. In *Proc. of INTERSPEECH*.

G. Murray, S. Renals, J. Carletta, and J. Moore. 2005b. Evaluating automatic summaries of meeting recordings. In *Proc. of ACL 2005 MTSE Workshop*.

E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, and H. Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proc. of SIGdial Workshop on Discourse and Dialogue*.

J. Turner and E. Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proc. of ACL*.

S. Xie, Y. Liu, and H. Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *Proc. of IEEE Workshop on Spoken Language Technology*.

[6] Thanks to Shasha Xie for generating these results.

Automatic Story Segmentation using a Bayesian Decision Framework for Statistical Models of Lexical Chain Features

Wai-Kit Lo
The Chinese University of Hong Kong, Hong Kong, China
wklo@se.cuhk.edu.hk

Wenying Xiong
The Chinese University of Hong Kong, Hong Kong, China
wyxiong@se.cuhk.edu.hk

Helen Meng
The Chinese University of Hong Kong, Hong Kong, China
hmmeng@se.cuhk.edu.hk

Abstract

This paper presents a Bayesian decision framework that performs automatic story segmentation based on statistical modeling of one or more lexical chain features. Automatic story segmentation aims to locate the instances in time where a story ends and another begins. A lexical chain is formed by linking coherent lexical items chronologically. A story boundary is often associated with a significant number of lexical chains ending before it, starting after it, as well as a low count of chains continuing through it. We devise a Bayesian framework to capture such behavior, using the lexical chain features of start, continuation and end. In the scoring criteria, lexical chain starts/ends are modeled statistically with the Weibull and uniform distributions at story boundaries and non-boundaries respectively. The normal distribution is used for lexical chain continuations. Full combination of all lexical chain features gave the best performance (F1=0.6356). We found that modeling chain continuations contributes significantly towards segmentation performance.

1 Introduction

Automatic story segmentation is an important precursor in processing audio or video streams in large information repositories. Very often, these continuous streams of data do not come with boundaries that segment them into semantically coherent units, or stories. The story unit is needed for a wide range of spoken language information retrieval tasks, such as topic tracking, clustering, indexing and retrieval. To perform automatic story segmentation, there are three categories of cues available: lexical cues from transcriptions, prosodic cues from the audio stream and video cues such as anchor face and color histograms. Among the three types of cues, lexical cues are the most generic since they can work on text and multimedia sources. Previous approaches include TextTiling (Hearst 1997) that monitors changes in sentence similarity, use of cue phrases (Reynar 1999) and Hidden Markov Models (Yamron 1998). In addition, the approach based on lexical chaining captures the content coherence by linking coherent lexical items (Morris and Hirst 1991, Hirst and St-Onge 1998). Stokes (2004) discovers boundaries by chaining up terms and locating instances of time where the count of chain starts and ends (boundary strength) achieves local maxima. Chan *et al.* (2007) enhanced this approach through statistical modeling of lexical chain starts and ends. We further extend this approach in two aspects: 1) a Bayesian decision framework is used; 2) chain continuations straddling across boundaries are taken into consideration and statistically modeled.

2 Experimental Setup

Experiments are conducted using data from the TDT-2 Voice of America Mandarin broadcast. In particular, we only use the data from the long programs (40 programs, 1458 stories in total), each of which is about one hour in duration. The average number of words per story is 297. The news programs are further divided chronologically into training (for parameter estimation of the statistical models), development (for tuning decision thresholds) and test (for performance evaluation) sets, as shown in Figure 1. Automatic speech recognition (ASR) outputs that are provided in the TDT-2 corpus are used for lexical chain formation.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 265–268,
Suntec, Singapore, 4 August 2009.

The story segmentation task in this work is to decide whether a hypothesized utterance boundary (provided in the TDT-2 data based on the speech recognition result) is a story boundary. Segmentation performance is evaluated using the F1-measure.

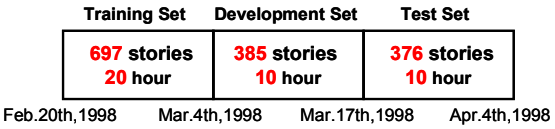


Figure 1: Organization of the long programs in TDT-2 VOA Mandarin for our experiments.

3 Approach

Our approach considers utterance boundaries that are labeled in the TDT-2 corpus and classifies them either as a story boundary or non-boundary.

We form lexical chains from the TDT-2 ASR outputs by linking repeated words. Since words may also repeat across different stories, we limit the maximum distance between consecutive words within the lexical chain. This limit is optimized according to the approach in (Chan *et al.* 2007) based on the training data. The optimal value is found to be 130.9sec for long programs.

We make use of three lexical chain features: chain starts, continuations and ends. At the beginning of a story, new words are introduced more frequently and hence we observe many lexical chain starts. There is also tendency of many lexical chains ending before a story ends. As a result, there is a higher density of chain starts and ends in the proximity of a story boundary. Furthermore, there tends to be fewer chains straddling across a story boundary. Based on these characteristics of lexical chains, we devise a statistical framework for story segmentation by modeling the distribution of these lexical chain features near the story boundaries.

3.1 Story Segmentation based on a Single Lexical Chain Feature

Given an utterance boundary with the lexical chain feature, X, we compare the conditional probabilities of observing a boundary, B, or non-boundary, $\overline{B}$, as

$$P(B \mid X) \gtrless P(\overline{B} \mid X). \quad (1)$$

where X is a single chain feature, which may be the chain start (S), chain continuation (C), *or* chain end (E).

By applying the Bayes' theorem, this can be rewritten as a likelihood ratio test,

$$\frac{P(X \mid B)}{P(X \mid \overline{B})} \gtrless \theta_x \quad (2)$$

for which the decision threshold is $\theta_x = P(\overline{B})/P(B)$, dependent on the a priori probability of observing boundary or a non-boundary.

3.2 Story Segmentation based on Combined Chain Features

When multiple features are used in combination, we formulate the problem as

$$P(B \mid S,E,C) \gtrless P(\overline{B} \mid S,E,C). \quad (3)$$

By assuming that the chain features are conditionally independent of one another (i.e., $P(S,C,E|B) = P(S|B)\,P(C|B)\,P(E|B)$), the formulation can be rewritten as a likelihood ratio test

$$\frac{P(S \mid B)P(E \mid B)P(C \mid B)}{P(S \mid \overline{B})P(E \mid \overline{B})P(C \mid \overline{B})} \gtrless \theta_{SEC}. \quad (4)$$

4 Modeling of Lexical Chain Features

4.1 Chain starts and ends

We follow (Chan *et al.* 2007) to model the lexical chain starts and ends at a story boundary with a statistical distribution. We apply a window around the candidate boundaries (same window size for both chain starts and ends) in our work. Chain features falling outside the window are excluded from the model. Figure 2 shows the distribution when a window size of 20 seconds is used. This is the optimal window size when chain start and end features are combined.

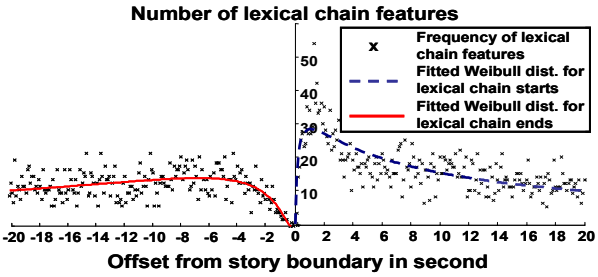


Figure 2: Distribution of chain starts and ends at known story boundaries. The Weibull distribution is used to model these distributions.

We also assume that the probability of seeing a lexical chain start / end at a particular instance is independent of the starts / ends of other chains. As a result, the probability of seeing a sequence of chain starts at a story boundary is given by the product of a sequence of Weibull distributions

$$P(S \mid B) = \prod_{i=1}^{N_s} \frac{k}{\lambda}\left(\frac{t_i}{\lambda}\right)^{k-1} e^{-\left(\frac{t_i}{\lambda}\right)^k}, \quad (5)$$

where S is the sequence of time with chain starts ($S=[t_1, t_2, \ldots t_i, \ldots t_{Ns}]$), k_s is the shape, λ_s is the scale for the fitted Weibull distribution for chain starts, N_s is the number of chain starts. The same formulation is similarly applied to chain ends.

Figure 3 shows the frequency of raw feature points for lexical chain starts and ends near utterance boundaries that are non-story boundaries. Since there is no obvious distribution pattern for these lexical chain features near a non-story boundary, we model these characteristics with a uniform distribution.

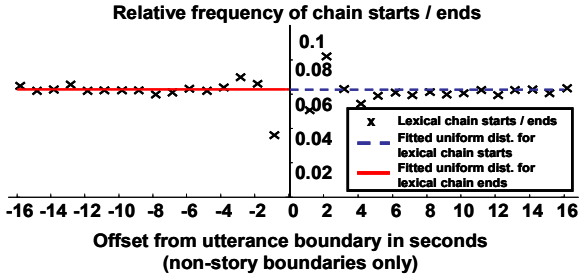


Figure 3: Distribution of chain starts and ends at utterance boundaries that are non-story boundaries.

4.2 Chain continuations

Figure 4 shows the distributions of chain continuations near story boundary and non-story boundary. As one may expect, there are fewer lexical chains that straddle across a story boundary (the curve of $P(C \mid B)$) when compared to a non-story boundary (the curve of $P(C \mid \overline{B})$). Based on the observations, we model the probability of occurrence of lexical chains straddling across a given story boundary or non-story boundary by a normal distribution.

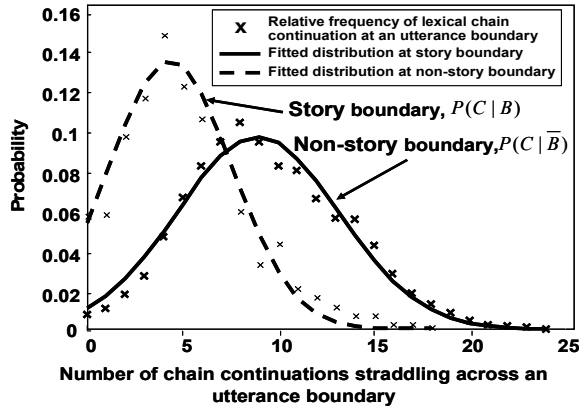


Figure 4: Distributions of chain continuations at story boundaries and non-story boundaries.

5 Story Segmentation based on Combination of Lexical Chain Features

We trained the parameters of the Weibull distribution for lexical chain starts and ends at story boundaries, the uniform distribution for lexical chain start / end at non-story boundary, and the normal distribution for lexical chain continuations. Instead of directly using a threshold as shown in Equation (2), we optimize on the parameter n, which is the optimal number of top scoring utterance boundaries that are classified as story boundaries in the development set.

5.1 Using Bayesian decision framework

We compare the performance of the Bayesian decision framework to the use of likelihood only $P(X|B)$ as shown in Figure 5. The results demonstrate consistent improvement in F1-measure when using the Bayesian decision framework.

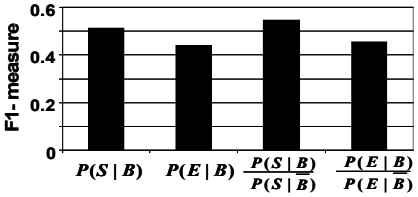


Figure 5: Story segmentation performance in F1-measure when using single lexical chain features.

5.2 Modeling multiple features jointly

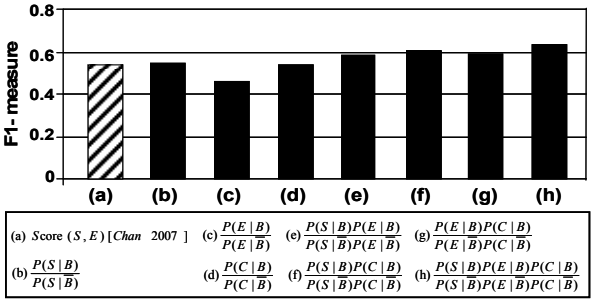


Figure 6: Results of F1-measure comparing the segmentation results using different statistical models of lexical chain features.

We further compare the performance of various scoring methods including single and combined lexical chain features. The baseline result is obtained using a scoring function based on the likelihoods of seeing a chain start or end at a story boundary (Chan *et al.* 2007) which is denoted as *Score(S, E)*. Performance from other methods based on the same dataset can be referenced from Chan *et al.* 2007 and will not be repeated here. The best story segmentation performance is achieved by combining all lexical chain features which achieves an F1-measure of 0.6356. All improvements have been verified to be statistically significant (α=0.05). By comparing the results of (e) to (h), (c) to (g), and (b) to (f), we can see that lexical chain continuation feature contributes significantly and consistently towards story segmentation performance.

5.3 Analysis

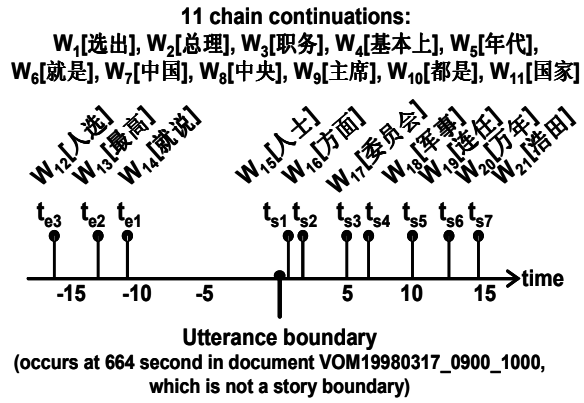


Figure 7: Lexical chain starts, ends and continuations in the proximity of a non-story boundary. W_i[xxxx] denotes the *i*-th Chinese word "xxxx".

Figure 7 shows an utterance boundary that is a non-story boundary. There is a high concentration of chain starts and ends near the boundary which leads to a misclassification if we only combine chain starts and ends for segmentation. However, there are also a large number of chain continuations across the utterance boundary, which implies that a story boundary is less likely. The full combination gives the correct decision.

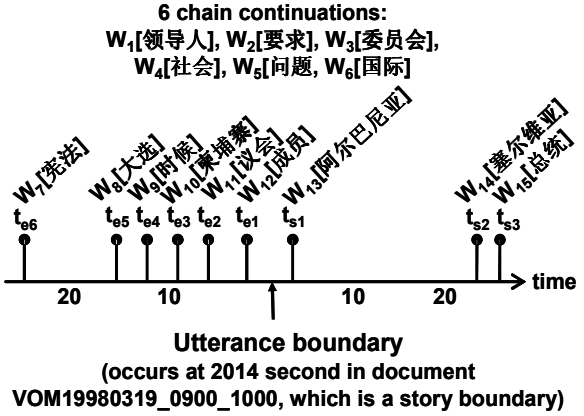


Figure 8: Lexical chain starts, ends and continuations in the proximity of a story boundary.

Figure 8 shows another example where an utterance boundary is misclassified as a non-story boundary when only the combination of lexical chain starts and ends are used. Incorporation of the chain continuation feature helps rectify the classification.

From these two examples, we can see that the incorporation of chain continuation in our story segmentation framework can complement the features of chain starts and ends. In both examples above, the number of chain continuations plays a crucial role in correct identification of a story boundary.

6 Conclusions

We have presented a Bayesian decision framework that performs automatic story segmentation based on statistical modeling of one or more lexical chain features, including lexical chain starts, continuations and ends. Experimentation shows that the Bayesian decision framework is superior to the use of likelihoods for segmentation. We also experimented with a variety of scoring criteria, involving likelihood ratio tests of a single feature (i.e. lexical chain starts, continuations or ends), their pair-wise combinations, as well as the full combination of all three features. Lexical chain starts/ends are modeled statistically with the Weibull and normal distributions for story boundaries and non-boundaries. The normal distribution is used for lexical chain continuations. Full combination of all lexical chain features gave the best performance (F1=0.6356). Modeling chain continuations contribute significantly towards segmentation performance.

Acknowledgments

This work is affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies. We would also like to thank Professor Mari Ostendorf for suggesting the use of continuing chains and Mr. Kelvin Chan for providing information about his previous work.

References

Chan, S. K. *et al*. 2007. "Modeling the Statistical Behaviour of Lexical Chains to Capture Word Cohesiveness for Automatic Story Segmentation", *Proc. of INTERSPEECH-2007*.

Hearst, M. A. 1997. "TextTiling: Segmenting Text into Multiparagraph Subtopic Passages", *Computational Linguistics*, 23(1), pp. 33–64.

Hirst, G. and St-Onge, D. 1998. "Lexical chains as representations of context for the detection and correction of malapropisms", *WordNet: An Electronic Lexical Database*, pp. 305–332.

Morris, J. and Hirst, G. 1991. "Lexical cohesion computed by thesaural relations as an indicator of the structure of text", *Computational Linguistics*, 17(1), pp. 21–48.

Reynar, J.C. 1999, "Statistical models for topic segmentation", *Proc. 37th annual meeting of the ACL*, pp. 357–364.

Stokes, N. 2004. Applications of Lexical Cohesion Analysis in the Topic Detection and Tracking Domain, PhD thesis, University College Dublin.

Yamron, J.P. *et al.* 1998, "A hidden Markov model approach to text segmentation and event tracking", *Proc. ICASSP 1998*, pp. 333–336.

Investigating Pitch Accent Recognition in Non-native Speech

Gina-Anne Levow
Computer Science Department
University of Chicago
ginalevow@gmail.com

Abstract

Acquisition of prosody, in addition to vocabulary and grammar, is essential for language learners. However, it has received less attention in instruction. To enable automatic identification and feedback on learners' prosodic errors, we investigate automatic pitch accent labeling for non-native speech. We demonstrate that an acoustic-based context model can achieve accuracies over 79% on binary pitch accent recognition when trained on within-group data. Furthermore, we demonstrate that good accuracies are achieved in cross-group training, where native and near-native training data result in no significant loss of accuracy on non-native test speech. These findings illustrate the potential for automatic feedback in computer-assisted prosody learning.

1 Introduction

Acquisition of prosody, in addition to vocabulary and grammar, is essential for language learners. However, intonation has been less-emphasized both in classroom and computer-assisted language instruction (Chun, 1998). Outside of tone languages, it can be difficult to characterize the factors that lead to non-native prosody in learner speech, and it is difficult for instructors to find time for the one-on-one interaction that is required to provide feedback and instruction in prosody.

To address these problems and enable automatic feedback to learners in a computer-assisted language learning setting, we investigate automatic prosodic labelling of non-native speech. While many prior systems (Teixeia et al., 2000; Tepperman and Narayanan, 2008) aim to assign a score to the learner speech, we hope to provide more focused feedback by automatically identifying prosodic units, such as pitch accents in English or tone in Mandarin, to enable direct comparison with gold-standard native utterances.

There has been substantial progress in automatic pitch accent recognition for native speech, achieving accuracies above 80% for acoustic-feature based recognition in multi-speaker corpora (Sridhar et al., 2007; Levow, 2008). However, there has been little study of pitch accent recognition in non-native speech. Given the challenges posed for automatic speech recognition of non-native speech, we ask whether recognition of intonational categories is practical for non-native speech. To lay the foundations for computer-assisted intonation tutoring, we ask whether competitive accuracies can be achieved on non-native speech. We further investigate whether good recognition accuracy can be achieved using relatively available labeled native or near-native speech, or whether it will be necessary to collect larger amounts of training or adaptation data matched for speaker, language background, or language proficiency.

We employ a pitch accent recognition approach that exploits local and coarticulatory context to achieve competitive pitch accent recognition accuracy on native speech. Using a corpus of prosodically labelled native and non-native speech, we illustrate that similar acoustic contrasts hold for pitch accents in both native and non-native speech. These contrasts yield competitive accuracies on binary pitch accent recognition using within-group training data. Furthermore, there is no significant drop in accuracy when models trained on native or near-native speech are employed for classification of non-native speech.

The remainder of the paper is organized as follows. We present the LeaP Corpus used for our experiments in Section 2. We next describe the feature sets employed for classification (Section 3) and contrastive acoustic analysis for these features in native and non-native speech (Section 4). We

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 269–272,
Suntec, Singapore, 4 August 2009.

ID	Description
c1	non-native, before prosody training
c2	non-native, after first prosody training
c3	non-native, after second prosody training
e1	non-native, before going abroad
e2	non-native, after going abroad
sl	'super-learner', near-native
na	native

Table 1: Speaker groups, with ID and description in the LeaP Corpus

then describe the classifier setting and experimental results in Section 5 as well as discussion. Finally, we present some conclusions and plans for future work.

2 LeaP Corpus and the Dataset

We employ data from the LeaP Corpus (Milde and Gut, 2002), collected at the University of Bielefeld as part of the "Learning Prosody in a Foreign Language" project. Details of the corpus (Milde and Gut, 2002), inter-rater reliability measures (Gut and Bayerl, 2004), and other research findings (Gut, 2009) have been reported.

Here we focus on the read English segment of the corpus that has been labelled with modified EToBI tags[1] , to enable better comparison with prior results of prosodic labelling accuracy and also to better model a typical language laboratory setting where students read or repeat. This yields a total of 37 recordings of just over 300 syllables each, from 26 speakers, as in Table 1.[2] This set allows the evaluation of prosodic labelling across a range of native and non-native proficiency levels. The modified version of EToBI employed by the LeaP annotators allows transcription of 14 categories of pitch accent and 14 categories of boundary tone. However, in our experiments, we will focus only on pitch accent recognition and will collapse the inventory to the relatively standard, and more reliably annotated, four-way (high, downstepped high, low, and unaccented) and binary (accented, unaccented) label sets.

[1]While the full corpus includes speakers from a range of languages, the EToBI labels were applied primarily to data from German speakers.

[2]Length of recordings varies due to differences in syllabification and cliticization, as well as disfluencies and reading errors.

3 Acoustic-Prosodic Features

Recent research has highlighted the importance of context for both tone and intonation. The role of context can be seen in the characterization of pitch accents such as down-stepped high and in phenomena such as downdrift across a phrase. Further, local coarticulation with neighboring tones has been shown to have a significant impact on the realization of prosodic elements, due to articulatory constraints (Xu and Sun, 2002). The use of prosodic and coarticulatory context has improved the effectiveness of tone and pitch accent recognition in a range of languages (Mandarin (Wang and Seneff, 2000), English (Sun, 2002)) and learning frameworks (decision trees (Sun, 2002), HMMs (Wang and Seneff, 2000), and CRFs (Levow, 2008)).

Thus, in this work, we employ a rich contextual feature set, based on that in (Levow, 2008). We build on the pitch target approximation model, taking the syllable as the domain of tone prediction with a pitch height and contour target approached exponentially over the course of the syllable, consistent with (Sun, 2002). We employ an acoustic model at the syllable level, employing pitch, intensity and duration measures. The acoustic measures are computed using Praat's (Boersma, 2001) "To pitch" and "To intensity." We log-scaled and speaker-normalized all pitch and intensity values.

We compute two sets of features: one set describing features local to the syllable and one set capturing contextual information.

3.1 Local features

We extract features to represent the pitch height and pitch contour of the syllable. For pitch features, we extract the following information: (a) pitch values for five evenly spaced points in the voiced region of the syllable, (b) pitch maximum, mean, minimum, and range, and (c) pitch slope, from midpoint to end of syllable. We also obtain the following non-pitch features: (a) intensity maximum and mean and (b) syllable duration.

3.2 Context Modeling

To capture local contextual influences and cues, we employ two sets of features. The first set of features includes differences between pitch maxima, pitch means, pitches at the midpoint of the syllables, pitch slopes, intensity maxima, and intensity means, between the current and preceding or fol-

lowing syllable. The second set of features adds the last pitch values from the end of the preceding syllable and the first from the beginning of the following syllable. These features capture both the relative differences in pitch associated with pitch accent as well as phenomena such as pitch peak delay in which the actual pitch target may not be reached until the following syllable.

4 Acoustic Analysis of Native and Non-native Tone

To assess the potential effectiveness of tone recognition for non-native speech, we analyze and compare native and non-native speech with respect to features used for classification that have shown utility in prior work. Pitch accents are characterized not only by their absolute pitch height, but also by contrast with neighboring syllables. Thus, we compare the values for pitch and delta pitch, the difference between the current and preceding syllable, both with log-scaled measures for high-accented and unaccented syllables. We contrast these values within speaker group (native: na; non-native: e1, c1). We also compare the delta pitch measures between speaker groups (na versus e1 or c1).

Not only do we find significant differences for delta pitch between accented and unaccented syllables for native speakers as we expect, but we find that non-native speakers also exhibit significant differences for this measure (t-test, two-tailed,$p < 0.001$). Accented syllables are reliably higher in pitch than immediately preceding syllables, while unaccented syllables show no contrast. Importantly, we further observe a significant difference in delta pitch for high accented syllables between native and non-native speech. Native speakers employ a markedly larger change in pitch to indicate accent than do non-native speakers, a fine-grained view consistent with findings that non-native speakers employ a relatively compressed pitch range (Gut, 2009).

For one non-native group (e1), we find that although these speakers produce reliable contrasts in delta pitch between *neighboring* syllables, the overall pitch height of high accented syllables is not significantly different from that of unaccented syllables. For native speakers and the 'c1' non-native group, though, overall pitch height does differ significantly between accented and unaccented syllables. This finding suggests that while all speakers in this data set understand the locally contrastive role of pitch accent, some non-native speaker groups do not have as reliable global control of pitch.

The presence of these reliable contrasts between accented and unaccented syllables in both native and non-native speech suggests that automatic pitch accent recognition in learner speech could be successful.

5 Pitch Accent Recognition Experiments

We assess the effectiveness of pitch accent recognition on the LeaP Corpus speech. We hope to understand whether pitch accent can be accurately recognized in non-native speech and whether accuracy rates would be competitive with those on native speech. In addition, we aim to compare the impact of different sources of training data. We assess whether non-native prosody can be recognized using native or near-native training speech or whether it will be necessary to use matched training data from non-natives of similar skill level or language background.

Thus we perform experiments on matched training and test data, training and testing within groups of speakers. We also evaluate cross-group training and testing, training on one group of speakers (native and near-native) and testing on another (non-native). We contrast all these results with assignment of the dominant 'unaccented' label to all instances (common class).

5.1 Support Vector Machine Classifier

For all supervised experiments reported in this paper, we employ a Support Vector machine (SVM) with a linear kernel. Support Vector Machines provide a fast, easily trainable classification framework that has proven effective in a wide range of application tasks. For example, in the binary classification case, given a set of training examples presented as feature vectors of length D, the linear SVM algorithm learns a vector of weights of length D which is a linear combination of a subset of the input vectors and performs classification based on the function $f(x) = sign(w^T x - b)$. We employ the publicly available implementation of SVMs, LIBSVM (C-C.Cheng and Lin, 2001).

5.2 Results

We see that, for within group training, on the binary pitch accent recognition task, accuracies

	c1	c2	c3	e1	e2	*sl*	**na**
Within-group Accuracy	79.1	80.9	80.6	81	82.5	*82.4*	**81.2**
Cross-group Accuracy (**na**)	77.2	79	81.4	80.3	82.5	*83.2*	
Cross-group Accuracy (*sl*)	77.3	79.9	82	80.5	82.9		**81.6**
Common Class	56.9	59.6	56.2	70.2	64	*65.5*	**63.6**

Table 2: Pitch accent recognition, within-group, cross-group with native and near-native training, and most common class baseline: Non-native (plain), 'Super-learner' (underline *sl*), Native (bold **na**)

range from approximately 79% to 82.5%. These levels are consistent with syllable-, acoustic-feature-based prosodic recognition reported in the literature (Levow, 2008). A summary of these results appears in Table 2. In the cross-group training and testing condition, we observe some variations in accuracy, for some training sets. However, crucially none of the differences between native-based or near-native training and within-group training reach significance for the binary pitch accent recognition task.

6 Conclusion

We have demonstrated the effectiveness of pitch accent recognition on both native and non-native data from the LeaP corpus, based on significant differences between accented and unaccented syllables in both native and non-native speech. Although these differences are significantly larger in native speech, recognition remains robust to training with native speech and testing on non-native speech, without significant drops in accuracy. This result argues that binary pitch accent recognition using native training data may be sufficiently accurate that to avoid collection and labeling of large amounts of training data matched by speaker or fluency-level to support prosodic annotation and feedback. In future work, we plan to incorporate prosodic recognition and synthesized feedback to support computer-assisted prosody learning.

Acknowledgments

We thank the creators of the LeaP Corpus as well as C-C. Cheng and C-J. Lin for LibSVM. This work was supported by NSF IIS: 0414919.

References

P. Boersma. 2001. Praat, a system for doing phonetics by computer. *Glot International*, 5(9–10):341–345.

C-C.Cheng and C-J. Lin. 2001. LIBSVM:a library for support vector machines. Software available at: http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Dorothy M. Chun. 1998. Signal analysis software for teaching discourse intonation. *Language Learning & Technology*, 2(1):61–77.

U. Gut and P. S. Bayerl. 2004. Measuring the reliability of manual annotations of speech corpora. In *Proceedings of Speech Prosody 2004*.

U. Gut. 2009. *Non-native speech. A corpus-based analysis of phonological and phonetic properties of L2 English and German*. Peter Lang, Frankfurt.

G.-A. Levow. 2008. Automatic prosodic labeling with conditional random fields and rich acoustic features. In *Proceedings of the IJCNLP 2008*.

J.-T. Milde and U. Gut. 2002. A prosodic corpus of non-native speech. In *Proceedings of the Speech Prosody 2002 Conference*, pages 503–506.

V. Rangarajan Sridhar, S. Bangalore, and S. Narayanan. 2007. Exploiting acoustic and syntactic features for prosody labeling in a maximum entropy framework. In *Proceedings of HLT NAACL 2007*, pages 1–8.

Xuejing Sun. 2002. Pitch accent prediction using ensemble machine learning. In *Proceedings of ICSLP-2002*.

C. Teixeia, H. Franco, E. Shriberg, K. Precoda, and K. Somnez. 2000. Prosodic features for automatic text-independent evaluation of degree of nativeness for language learners. In *Proceedings of ICSLP 2000*.

J. Tepperman and S. Narayanan. 2008. Better non-native intonation scores through prosodic theory. In *Proceedings of Interspeech 2008*.

C. Wang and S. Seneff. 2000. Improved tone recognition by normalizing for coarticulation and intonation effects. In *Proceedings of 6th International Conference on Spoken Language Processing*.

Yi Xu and X. Sun. 2002. Maximum speed of pitch change and how it may relate to speech. *Journal of the Acoustical Society of America*, 111.

A Stochastic Finite-State Morphological Parser for Turkish

Haşim Sak & Tunga Güngör
Dept. of Computer Engineering
Boğaziçi University
TR-34342, Bebek, İstanbul, Turkey
hasim.sak@boun.edu.tr
gungort@boun.edu.tr

Murat Saraçlar
Dept. of Electrical & Electronics Engineering
Boğaziçi University
TR-34342, Bebek, İstanbul, Turkey
murat.saraclar@boun.edu.tr

Abstract

This paper presents the first stochastic finite-state morphological parser for Turkish. The non-probabilistic parser is a standard finite-state transducer implementation of two-level morphology formalism. A disambiguated text corpus of 200 million words is used to stochastize the morphotactics transducer, then it is composed with the morphophonemics transducer to get a stochastic morphological parser. We present two applications to evaluate the effectiveness of the stochastic parser; spelling correction and morphology-based language modeling for speech recognition.

1 Introduction

Turkish is an agglutinative language with a highly productive inflectional and derivational morphology. The computational aspects of Turkish morphology have been well studied and several morphological parsers have been built (Oflazer, 1994), (Güngör, 1995).

In language processing applications, we may need to estimate a probability distribution over all word forms. For example, we need probability estimates for unigrams to rank misspelling suggestions for spelling correction. None of the previous studies for Turkish have addressed this problem. For morphologically complex languages, estimating a probability distribution over a static vocabulary is not very desirable due to high out-of-vocabulary rates. It would be very convenient for a morphological parser as a word generator/analyzer to also output a probability estimate for a word generated/analyzed. In this work, we build such a stochastic morphological parser for Turkish[1] and give two example applications for evaluation.

[1]The stochastic morphological parser is available for research purposes at http://www.cmpe.boun.edu.tr/~hasim

2 Language Resources

We built a morphological parser using the two-level morphology formalism of Koskenniemi (1984). The two-level phonological rules and the morphotactics were adapted from the PC-KIMMO implementation of Oflazer (1994). The rules were compiled using the *twolc* rule compiler (Karttunen and Beesley, 1992). A new root lexicon of 55,278 words based on the Turkish Language Institution dictionary[2] was compiled. For finite-state operations and for running the parser, we used the OpenFST weighted finite-state transducer library (Allauzen et al., 2007). The parser can analyze about 8700 words per second on a 2.33 GHz Intel Xeon processor.

We need a text corpus for estimating the parameters of a statistical model of morphology. For this purpose, we compiled a text corpus of 200 million-words by collecting texts from online newspapers. The morphological parser can analyze about 96.7% of the tokens.

The morphological parser may output more than one possible analysis for a word due to ambiguity. For example, the parser returns four analyses for the word *kedileri* as shown below. The morphological representation is similar to the one used by Oflazer and Inkelas (2006).

kedi[Noun]+*lAr*[A3pl]+*SH*[P3sg]+[Nom] (his/her cats)
kedi[Noun]+*lAr*[A3pl]+[Pnon]+*YH*[Acc] (the cats)
kedi[Noun]+*lAr*[A3pl]+*SH*[P3pl]+[Nom] (their cats)
kedi[Noun]+[A3sg]+*lArH*[P3pl]+[Nom] (their cat)

We need to resolve this ambiguity to train a probabilistic morphology model. For this purpose, we used our averaged perceptron-based morphological disambiguator (Sak et al., 2008). The disambiguation system achieves about 97.05% disambiguation accuracy on the test set.

[2]http://www.tdk.gov.tr

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 273–276,
Suntec, Singapore, 4 August 2009.

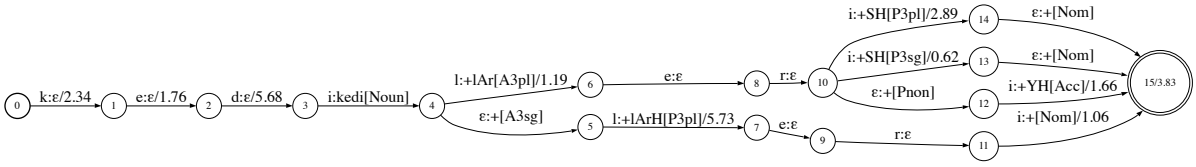


Figure 1: Finite-state transducer for the word *kedileri*.

3 Stochastic Morphological Parser

The finite-state transducer of the morphological parser is obtained as the composition of the morphophonemics transducer mp and the morphotactics transducer mt; $mp \circ mt$. The morphotactics transducer encodes the morphosyntax of the language. If we can estimate a statistical morphosyntactic model, we can convert the morphological parser to a probabilistic one by composing the probabilistic morphotactics transducer with the morphophonemics transducer. Eisner (2002) gives a general EM algorithm for parameter estimation in probabilistic finite-state transducers. The algorithm uses a bookkeeping trick (expectation semiring) to compute the expected number of traversals of each arc in the E step. The M step reestimates the probabilities of the arcs from each state to be proportional to the expected number of traversals of each arc - the arc probabilities are normalized at each state to make the finite-state transducer Markovian. However, we do not need this general method of training. Since we can disambiguate the possible morphosyntactic tag sequences of a word, there is a single path in the morphotactics transducer that matches the chosen morphosyntactic tag sequence. Then the maximum-likelihood estimates of the weights of the arcs in the morphotactics transducer are found by setting the weights proportional to the number of traversals of each arc. We can use a specialized semiring to cleanly and efficiently count the number of traversals of each arc.

Weights in finite-state transducers are elements of a semiring, which defines two binary operations $\otimes$ and $\oplus$, where $\otimes$ is used to combine the weights of arcs on a path into a path weight and $\oplus$ is used to combine the weights of alternative paths (Berstel and Reutenauer, 1988). We define a counting semiring to keep track of the number of traversals of each arc. The weights in the mt transducer are converted to the counting semiring. In this semiring, the weigths are vectors of integers having dimension as the total number of arcs in the mt transducer. We number the arcs in the mt transducer and set the weight of the n^{th} arc as the n^{th} basis vector. The binary plus $\oplus$ and the times $\otimes$ operations of the counting semiring are defined as the sum of the weight vectors. Thus, the n^{th} value of the vector in the counting semiring just counts the appearances of the n^{th} arc of mt in a path.

To estimate the weights of the stochastic model of the mt transducer, we use the text corpus collected from the web. First we parse the words in the corpus to get all the possible analyses of the words. Then we disambiguate the morphological analyses of the words to select one of the morphosyntactic tag sequences x_i for each word. We build a finite-state transducer $\epsilon \times x_i$ that maps ϵ symbol to x_i in the counting semiring. The weights of this transducer are zero vectors having the same dimension as the mt transducer. Then the finite-state transducer $(\epsilon \times x_i) \circ (mt \times \epsilon)$ having all $\epsilon : \epsilon$ arcs can be minimized to get a one-state FST which has the weight vector that keeps the number of traversals of each arc in mt. The weight vector is accumulated for all the x_i morphosyntactic tag sequences in the corpus. The final accumulated weight vector is used to assign probabilities to each arc in the mt transducer proportional to the traversal count of the arc, hence resulting in the stochastic morphotactics transducer $\tilde{mt}$. We use add-one smoothing to prevent the arcs having zero probability. The $\tilde{mt}$ transducer is composed with the morphophonemics transducer mp to get a stochastic morphological parser.

The stochastic parser now returns probabilities with the possible analyses of a word. Figure 1 shows the weighted paths for the four possible analyses of the word *kedileri* as represented in the stochastic parser. The weights are negative log probabilities.

4 Spelling Correction

The productive morphology of Turkish allows one to generate very long words such as

ölümsüzleştirdiğimizden. Therefore, the detection and the correction of spelling errors by presenting the user with a ranked list of spelling suggestions are highly desired. There have been some previous studies for spelling checking (Solak and Oflazer, 1993) and spelling correction (Oflazer, 1996). However there has been no study to address the problem of ranking spelling suggestions. One can use a stochastic morphological parser to do spelling checking and correction, and present spelling suggestions ranked with the parser output probabilities. We assume that a word is misspelled if the parser fails to return an analysis of the word. Our method for spelling correction is to enumerate all the valid and invalid candidates that resemble the incorrect input word and filter the invalid ones with the morphological parser.

To enumerate the alternative spellings for a misspelled word, we generate all the words in one-character edit distance with the input word, where we consider one symbol insertion, deletion or substitution, or transposition of adjacent symbols. The Turkish alphabet includes six special letters (*ç*, *ğ*, *ı*, *ö*, *ş*, *ü*) that do not exist in English. These characters may not be supported in some keyboards and message transfer protocols; thus people frequently use their nearest ASCII equivalents (*c*, *g*, *i*, *o*, *s*, *u*, respectively) instead of the correct forms, e.g., spelling *nasılsın* as *nasilsin*. Therefore, in addition to enumerating words in one edit distance, we also enumerate all the words from which the misspelled word can be obtained by replacing these special Turkish characters with their ASCII counterparts. For instance, for the word *nasilsin*, the alternative spellings *nasılsin*, *nasilsın*, and *nasılsın* will also be generated.

Note that although the context is important for spelling correction, we use only unigrams. One can build a morpheme based language model to incorporate the context information. We also limited the edit distance to 1, but it is straightforward to allow longer edit distances. We can build a finite-state transducer to enumerate and represent efficiently all the valid and invalid word forms that can be obtained by these edit operations on a word. For example, the deletion of a character can be represented by the regular expression $\Sigma^*(\Sigma : \epsilon)\Sigma^*$ which can be compiled as a finite-state transducer, where Σ is the alphabet. The union of the transducers encoding one-edit distance operations and the restoration of the special Turkish characters is precompiled and optimized with determinization and minimization algorithms for efficiency. A misspelled input word transducer can be composed with the resulting transducer and in turn with the morphological parser to filter out the invalid word forms. The words with their estimated probabilities can be read from the output transducer and constitute the list of spelling suggestions for the word. The probabilities are used to rank the list to show to the user. We also handle the spelling errors where omission of a space character causes joining of two correct words by splitting the word into all combinations of two strings and checking if the string pieces are valid word forms. An example list of suggestions with the assigned negative log probabilities and their English glosses for the misspelled word *nasilsin* is given below.

nasılsın (14.2) (How are you), *nakilsin* (15.3) (You are a transfer), *nesilsin* (21.0) (You are a generation), *nasipsin* (21.2) (You are a share), *basilsin* (23.9) (You are a bacillus)

On a manually chosen test set containing 225 correct words which have relatively more complex morphology and 43 commonly misspelled words, the Precision and the Recall scores for the detection of spelling errors are 0.81 and 0.93, respectively.

5 Morphology-based Language Modeling

The closure of the transducer for the stochastic parser can be considered as a morphology-based unigram language model. Different than standard unigram word language models, this morphology-based model can assign probabilities to words not seen in the training corpus. It can also achieve lower out-of-vocabulary (OOV) rates than models that use a static vocabulary by employing a relatively smaller number of root words in the lexicon.

We compared the performances of the morphology-based unigram language model and the unigram word language model on a broadcast news transcription task. The acoustic model uses Hidden Markov Models (HMMs) trained on 183.8 hours of broadcast news speech data. The test set contains 3.1 hours of speech data (2,410 utterances). A text corpus of 1.2 million words from the transcriptions of the news recordings was used to train the stochastic parser as explained in Section 3 and unigram word language models.

We experimented with four different language

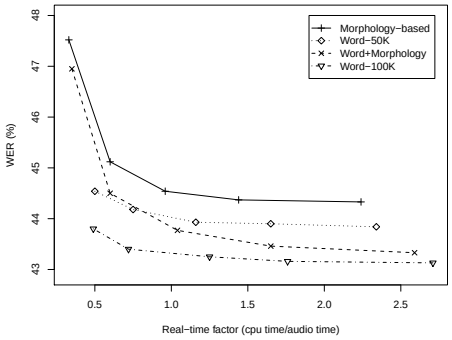


Figure 2: Word error rate versus real-time factor obtained by changing the pruning beam width.

models. Figure 2 shows the word error rate versus run-time factor for these models. In this figure the Word-50K and Word-100K are unigram word models with the specified vocabulary size and have the OOV rates 7% and 4.7% on the test set, respectively. The morphology-based model is based on the stochastic parser and has the OOV rate 2.8% . The 'word+morphology' model is the union of the morphology-based model and the unigram word model.

Even though the morphology-based model has a better OOV rate than the word models, the word error rate (WER) is higher. One of the reasons is that the transducer for the morphological parser is ambiguous and cannot be optimized for recognition in contrast to the word models. Another reason is that the probability estimates of this model are not as good as the word models since probability mass is distributed among ambiguous parses of a word and over the paths in the transducer. The 'word+morphology' model seems to alleviate most of the shortcomings of the morphology model. It performs better than 50K word model and is very close to the 100K word model. The main advantage of morphology-based models is that we have at hand the morphological analyses of the words during recognition. We plan to train a language model over the morphological features and use this model to rescore the hypothesis generated by the morphology-based models on-the-fly.

6 Conclusion

We described the first stochastic morphological parser for Turkish and gave two applications. The first application is a very efficient spelling correction system where probability estimates are used for ranking misspelling suggestions. We also gave the preliminary results for incorporating the morphology as a knowledge source in speech recognition and the results look promising.

Acknowledgments

This work was supported by the Boğaziçi University Research Fund under the grant numbers 06A102 and 08M103, the Scientific and Technological Research Council of Turkey (TÜBİTAK) under the grant number 107E261, the Turkish State Planning Organization (DPT) under the TAM Project, number 2007K120610 and TÜBİTAK BİDEB 2211.

References

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. OpenFst: A general and efficient weighted finite-state transducer library. In *CIAA 2007*, volume 4783 of *LNCS*, pages 11–23. Springer. `http://www.openfst.org`.

Jean Berstel and Christophe Reutenauer. 1988. *Rational Series and their Languages*. Springer-Verlag.

Jason Eisner. 2002. Parameter estimation for probabilistic finite-state transducers. In *ACL*, pages 1–8.

Tunga Güngör. 1995. *Computer Processing of Turkish: Morphological and Lexical Investigation*. Ph.D. thesis, Boğaziçi University.

Lauri Karttunen and Kenneth R. Beesley. 1992. Two-level rule compiler. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA.

Kimmo Koskenniemi. 1984. A general computational model for word-form recognition and production. In *ACL*, pages 178–181.

Kemal Oflazer and Sharon Inkelas. 2006. The architecture and the implementation of a finite state pronunciation lexicon for Turkish. *Computer Speech and Language*, 20(1):80–106.

Kemal Oflazer. 1994. Two-level description of Turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

Kemal Oflazer. 1996. Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, 22(1):73–89.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2008. Turkish language resources: Morphological parser, morphological disambiguator and web corpus. In *GoTAL 2008*, volume 5221 of *LNCS*, pages 417–427. Springer.

Ayşin Solak and Kemal Oflazer. 1993. Design and implementation of a spelling checker for turkish. *Literary and Linguistic Computing*, 8(3):113–130.

Parsing Speech Repair without Specialized Grammar Symbols*

Tim Miller
University of Minnesota
tmill@cs.umn.edu

Luan Nguyen
University of Minnesota
lnguyen@cs.umn.edu

William Schuler
University of Minnesota
schuler@cs.umn.edu

Abstract

This paper describes a parsing model for speech with repairs that makes a clear separation between linguistically meaningful symbols in the grammar and operations specific to speech repair in the operation of the parser. This system builds a model of how unfinished constituents in speech repairs are likely to finish, and finishes them probabilistically with placeholder structure. These modified repair constituents and the restarted replacement constituent are then recognized together in the same way that two coordinated phrases of the same type are recognized.

1 Introduction

Speech repair is a phenomenon in spontaneous spoken language in which a speaker decides to interrupt the flow of speech, replace some of the utterance (the "reparandum"), and continues on (with the "alteration") in a way that makes the whole sentence as transcribed grammatical only if the reparandum is ignored. As Ferreira et al. (2004) note, speech repairs[1] are the most disruptive type of disfluency, as they seem to require that a listener first incrementally build up syntactic and semantic structure, then subsequently remove it and rebuild when the repair is made. This difficulty combines with their frequent occurrence to make speech repair a pressing problem for machine recognition of spontaneous speech.

This paper introduces a model for dealing with one part of this problem, constructing a syntactic analysis based on a transcript of spontaneous spoken language. The model introduced here differs from other models attempting to solve the same problem, by completely separating the fluent grammar from the operations of the parser. The grammar thus has no representation of disfluency or speech repair, such as the "EDITED" category used to represent a reparandum in the Switchboard corpus, as such categories are seemingly at odds with the typical nature of a linguistic constituent.

Rather, the approach presented here uses a grammar that explicitly represents incomplete constituents being processed, and repair is represented by rules which allow incomplete constituents to be prematurely merged with existing structure. While this model is interesting for its elegance in representation, there is also reason to hypothesize improved performance, since this processing model requires no additional grammar symbols, and only one additional operation to account for speech repair, and thus makes better use of limited data resources.

2 Background

Previous work on parsing of speech with repairs has shown that syntactic cues can be used to increase accuracy of detection of reparanda, which can increase overall parsing accuracy. The first source of structure used to recognize repair is what Levelt (1983) called the "Well-formedness Rule." This rule essentially states that a speech repair acts like a conjunction; that is, the reparandum and the alteration must be of the same syntactic category. Of course, the reparandum is often unfinished, so the Well-formedness Rule allows for the reparandum category to be inferred.

This source of structure has been used by two related approaches, that of Hale et al. (2006) and Miller (2009). Hale and colleagues exploit this structure by adding contextual information to the standard reparandum label "EDITED". In their terminology, *daughter annotation* takes the (possibly unfinished) constituent label of the reparandum and appends it to the EDITED label. This

This research was supported by NSF CAREER award 0447685. The views expressed are not necessarily endorsed by the sponsors .

[1] Ferreira et al. use the term 'revisions'.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 277–280,
Suntec, Singapore, 4 August 2009.

allows a learned probabilistic context-free grammar to represent the likelihood of a reparandum of a certain type being a sibling with a finished constituent of the same type.

Miller's approach exploited the same source of structure, but changed the representation to use a REPAIRED label for alterations instead of an EDITED label for reparanda. The rationale for that change is the fact that a speech repair does not really begin until the interruption point, at which point the alteration is started and the reparandum is retroactively labelled as such. Thus, the argument goes, no special syntactic rules or symbols should be necessary until the alteration begins.

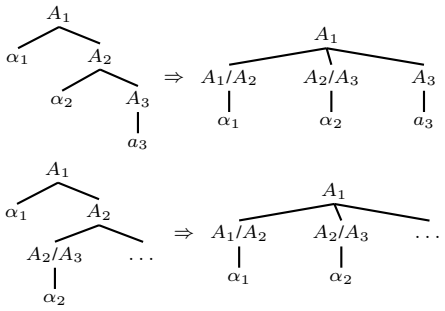
3 Model Description

3.1 Right-corner transform

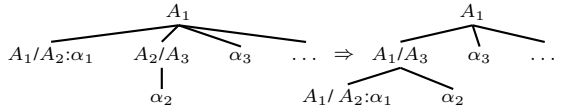
This work first uses a right-corner transform, which turns right-branching structure into left-branching structure, using category labels that use a "slash" notation α/γ to represent an incomplete constituent of type α "looking for" a constituent of type γ in order to complete itself.

This transform first requires that trees be binarized. This binarization is done in a similar way to Johnson (1998) and Klein and Manning (2003).

Rewrite rules for the right-corner transform are as follows, first flattening right-branching structure:[2]



then replacing it with left-branching structure:



One problem with this notation is the representation given to unfinished constituents, as seen in Figures 1 and 2. The standard representation of

[2]Here, all A_i denote nonterminal symbols, and α_i denote subtrees; the notation $A_1{:}\alpha_0$ indicates a subtree α_0 with label A_1; and all rewrites are applied recursively, from leaves to root.

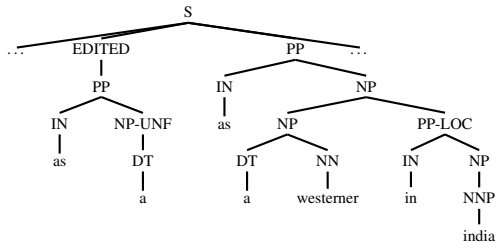


Figure 1: Section of interest of a standard phrase structure tree containing speech repair with unfinished noun phrase (NP).

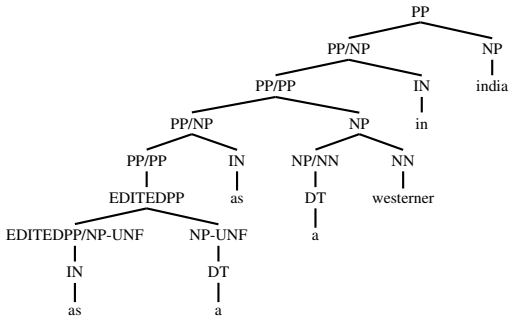


Figure 2: Right-corner transformed version of the fragment above. This tree requires several special symbols to represent the reparandum that starts this fragment.

an unfinished constituent in the Switchboard corpus is to append the -UNF label to the lowest unfinished constituent (see Figure 1). Since one goal of this work is separation of linguistic knowledge from language processing mechanisms, the -UNF tag should not be an explicit part of the grammar. In theory, the incomplete category notation induced by the right-corner transform is perfectly suited to this purpose. For instance, the category NP-UNF is a stand in category for several incomplete constituents, for example NP/NN, NP/NNS, etc. However, since the sub-trees with -UNF labels in the original corpus are by definition unfinished, the label to the right of the slash (NN in this case) is not defined. As a result, transformed trees with unfinished structure have the representation of Figure 2, which gives away the positive benefits of the right-corner transform in representing repair by propagating a special repair symbol (EDITED) through the grammar.

3.2 Approximating unfinished constituents

It is possible to represent -UNF categories as standard unfinished constituents, and account for unfinished constituents by having the parser prema-

turely end the processing of a given constituent. However, in the example given above, this would require predicting ahead of time that the NP-UNF was only missing a common noun – NN (for example). This problem is addressed in this work by probabilistically filling in placeholder final categories of unfinished constituents in the standard phrase structure trees, before applying the right-corner transform.

In order to fill in the placeholder with realistic items, phrase completions are learned from corpus statistics. First, this algorithm identifies an unfinished constituent to be finished as well as its existing children (in the continuing example, NP-UNF with child labelled DT). Next, the corpus is searched for fluent subtrees with matching root labels and child labels (NP and DT), and a distribution is computed of the actual completions of those subtrees. In the model used in this work, the most common completions are NN, NNS, and NNP. The original NP-UNF subtree is then given a placeholder completion by sampling from the distribution of completions computed above.

After this addition is complete, the UNF and EDITED labels are removed from the reparandum subtree, and if a restarted constituent of the same type is a sibling of the reparandum (e.g. another NP), the two subtrees are made siblings under a new subtree with the same category label (NP). See Figure 3 for a simple visual example of how this works.

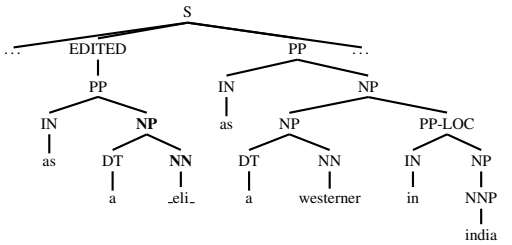


Figure 3: Same tree as in Figure 1, with the unfinished noun phrase now given a placeholder NN completion (both bolded).

Next, these trees are modified using the right-corner transform as shown in Figure 4. This tree still contains placeholder words that will not be in the text stream of an observed input sentence. Thus, in the final step of the preprocessing algorithm, the finished category label and the placeholder right child are removed where found in a right-corner tree. This results in a right-corner transformed tree in which a unary child or right

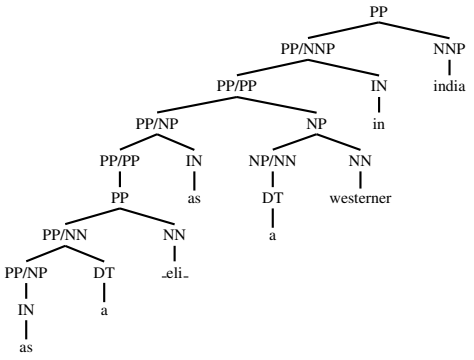


Figure 4: Right-corner transformed tree with placeholder finished phrase.

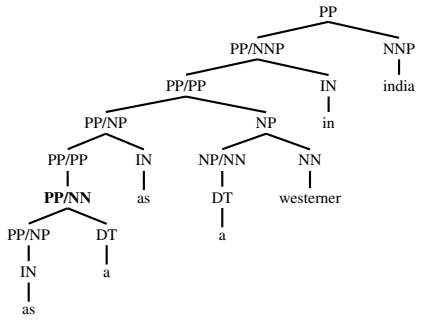


Figure 5: Final right-corner transformed state after excising placeholder completions to unfinished constituents. The bolded label indicates the signal of an unfinished category reparandum.

child subtree having an unfinished constituent type (a slash category, e.g. PP/NN in Figure 5) at its root represents a reparandum with an unfinished category. The tree then represents and processes the rest of the repair in the same way as a coordination.

4 Evaluation

This model was evaluated on the Switchboard corpus (Godfrey et al., 1992) of conversational telephone speech between two human interlocuters. The input to this system is the gold standard word transcriptions, segmented into individual utterances. For comparison to other similar systems, the system was given the gold standard part of speech for each input word as well. The standard train/test breakdown was used, with sections 2 and 3 used for training, and subsections 0 and 1 of section 4 used for testing. Several sentences from the end of section 4 were used during development.

For training, the data set was first standardized by removing punctuation, empty categories, typos, all categories representing repair structure,

and partial words – anything that would be difficult or impossible to obtain reliably with a speech recognizer.

The two metrics used here are the standard Parseval F-measure, and Edit-finding F. The first takes the F-score of labeled precision and recall of the non-terminals in a hypothesized tree relative to the gold standard tree. The second measure marks words in the gold standard as edited if they are dominated by a node labeled EDITED, and measures the F-score of the hypothesized edited words relative to the gold standard.

System Configuration	*Parseval-F*	*Edited-F*
Baseline CYK	71.05	18.03
Hale et al.	68.48	37.94
Plain RC Trees	69.07	30.89
Elided RC Trees	**67.91**	**24.80**
Merged RC Trees	**68.88**	**27.63**

Table 1: Results

Results of the testing can be seen in Table 1. The first line ("Baseline CYK") indicates the results using a standard probabilistic CYK parser, trained on the standardized input trees. The following two lines are results from reimplementations of the systems from Hale et al. (2006) and Miller (2009). The line marked 'Elided trees' gives current results. Surprisingly, this result proves to be lower than the previous results. Two observations in the output of the parser on the development set gave hints as to the reasons for this performance loss.

First, repairs using the slash categories (for unfinished reparanda) were rare (relative to finished reparanda). This led to the suspicion that there was a state-splitting phenomenon, where categories previously lumped together as EDITED-NP were divided into several unfinished categories (NP/NN, NP/NNS, etc.). To test this suspicion, another experiment was performed where all unary child and right child subtrees with unfinished category labels X/Y were replaced with EDITED-X. This result is shown in line five of Table 1. This result improves on the elided version, and suggests that the state-splitting effect is most likely one cause of decreased performance.

The second effect in the parser output was the presence of several very long reparanda (more than ten words), which are highly unlikely in normal speech. This phenomenon does not occur in the 'Plain RC Trees' condition. One explanation for this effect is that plain RC trees use the EDITED label in each rule of the reparandum (see Figure 2 for a short real-world example). This essentially creates a reparandum rule set, making expansion of a reparandum difficult due to the likelihood of a long chain eventually requiring a reparandum rule that was not found in the training data, or was not learned correctly in the much smaller set of reparandum-specific training data.

5 Conclusion and Future Work

In conclusion, this paper has presented a new model for speech containing repairs that enforces a clean separation between linguistic categories and parsing operations. Performance was below expectations, but analysis of the interesting reasons for these results suggests future directions. A model which explicitly represents the distance that a speaker backtracks when making a repair would prevent the parser from hypothesizing the unlikely reparanda of great length.

References

Fernanda Ferreira, Ellen F. Lau, and Karl G.D. Bailey. 2004. Disfluencies, language comprehension, and Tree Adjoining Grammars. *Cognitive Science*, 28:721–749.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proc. ICASSP*, pages 517–520.

John Hale, Izhak Shafran, Lisa Yung, Bonnie Dorr, Mary Harper, Anna Krasnyanskaya, Matthew Lease, Yang Liu, Brian Roark, Matthew Snover, and Robin Stewart. 2006. PCFGs with syntactic and prosodic indicators of speech repairs. In *Proceedings of the 45th Annual Conference of the Association for Computational Linguistics (COLING-ACL)*.

Mark Johnson. 1998. PCFG models of linguistic tree representation. *Computational Linguistics*, 24:613–632.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430.

Willem J.M. Levelt. 1983. Monitoring and self-repair in speech. *Cognition*, 14:41–104.

Tim Miller. 2009. Improved syntactic models for parsing speech with repairs. In *Proceedings of the North American Association for Computational Linguistics*, Boulder, CO.

Efficient Inference of CRFs for Large-Scale Natural Language Data

Minwoo Jeong[†*] **Chin-Yew Lin**[‡] **Gary Geunbae Lee**[†]
[†]Pohang University of Science & Technology, Pohang, Korea
[‡]Microsoft Research Asia, Beijing, China
[†]{stardust,gblee}@postech.ac.kr [‡]cyl@microsoft.com

Abstract

This paper presents an efficient inference algorithm of conditional random fields (CRFs) for large-scale data. Our key idea is to decompose the output label state into an active set and an inactive set in which most unsupported transitions become a constant. Our method unifies two previous methods for efficient inference of CRFs, and also derives a simple but robust special case that performs faster than exact inference when the active sets are sufficiently small. We demonstrate that our method achieves dramatic speedup on six standard natural language processing problems.

1 Introduction

Conditional random fields (CRFs) are widely used in natural language processing, but extending them to large-scale problems remains a significant challenge. For simple graphical structures (e.g. linear-chain), an exact inference can be obtained efficiently if the number of output labels is not large. However, for large number of output labels, the inference is often prohibitively expensive.

To alleviate this problem, researchers have begun to study the methods of increasing inference speeds of CRFs. Pal et al. (2006) proposed a Sparse Forward-Backward (SFB) algorithm, in which marginal distribution is compressed by approximating the true marginals using Kullback-Leibler (KL) divergence. Cohn (2006) proposed a Tied Potential (TP) algorithm which constrains the labeling considered in each feature function, such that the functions can detect only a relatively small set of labels. Both of these techniques efficiently compute the marginals with a significantly reduced runtime, resulting in faster training and decoding of CRFs.

This paper presents an efficient inference algorithm of CRFs which unifies the SFB and TP approaches. We first decompose output labels states into *active* and *inactive* sets. Then, the active set is selected by feasible heuristics and the parameters of the inactive set are held a constant. The idea behind our method is that not all of the states contribute to the marginals, that is, only a small group of the labeling states has sufficient statistics. We show that the SFB and the TP are special cases of our method because they derive from our unified algorithm with a different setting of parameters. We also present a simple but robust variant algorithm in which CRFs efficiently learn and predict large-scale natural language data.

[*]Parts of this work were conducted during the author's internship at Microsoft Research Asia.

2 Linear-chain CRFs

Many versions of CRFs have been developed for use in natural language processing, computer vision, and machine learning. For simplicity, we concentrate on linear-chain CRFs (Lafferty et al., 2001; Sutton and McCallum, 2006), but the generic idea described here can be extended to CRFs of any structure.

Linear-chain CRFs are conditional probability distributions over *label sequences* which are conditioned on *input sequences* (Lafferty et al., 2001). Formally, $\mathbf{x} = \{x_t\}_{t=1}^{T}$ and $\mathbf{y} = \{y_t\}_{t=1}^{T}$ are sequences of input and output variables. Respectively, where T is the length of sequence, $x_t \in \mathcal{X}$ and $y_t \in \mathcal{Y}$ where $\mathcal{X}$ is the finite set of the input observations and $\mathcal{Y}$ is that of the *output label* state space. Then, a first-order linear-chain CRF is defined as:

$$p_{\boldsymbol{\lambda}}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}), \qquad (1)$$

where Ψ_t is the local potential that denotes the factor at time t, and $\boldsymbol{\lambda}$ is the parameter vector. $Z(\mathbf{x})$ is a partition function which ensures the probabilities of all state sequences sum to one. We assume that the potentials factorize according to a set of observation features $\{\phi_k^1\}$ and transition features $\{\phi_k^2\}$, as follows:

$$\Psi_t(y_t, y_{t-1}, \mathbf{x}) = \Psi_t^1(y_t, \mathbf{x}) \cdot \Psi_t^2(y_t, y_{t-1}), \qquad (2)$$

$$\Psi_t^1(y_t, \mathbf{x}) = e^{\sum_k \lambda_k^1 \phi_k^1(y_t, \mathbf{x})}, \qquad (3)$$

$$\Psi_t^2(y_t, y_{t-1}) = e^{\sum_k \lambda_k^2 \phi_k^2(y_t, y_{t-1})}, \qquad (4)$$

where $\{\lambda_k^1\}$ and $\{\lambda_k^2\}$ are weight parameters which we wish to learn from data.

Inference is significantly challenging both in learning and decoding CRFs. Time complexity is $\mathcal{O}(T|\mathcal{Y}|^2)$ for exact inference (i.e., forward-backward and Viterbi algorithm) of linear-chain CRFs (Lafferty et al., 2001). The inference process is often prohibitively expensive

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 281–284,
Suntec, Singapore, 4 August 2009.

when $|\mathcal{Y}|$ is large, as is common in large-scale tasks. This problem can be alleviated by introducing approximate inference methods based on reduction of the search spaces to be explored.

3 Efficient Inference Algorithm

3.1 Method

The key idea of our proposed efficient inference method is that the output label state $\mathcal{Y}$ can be decomposed to an *active* set $\mathcal{A}$ and an *inactive* set $\mathcal{A}^c$. Intuitively, many of the possible transitions $(y_{t-1} \rightarrow y_t)$ do not occur, or are unsupported, that is, only a small part of the possible labeling set is informative. The inference algorithm need not precisely calculate marginals or maximums (more generally, messages) for unsupported transitions. Our efficient inference algorithm approximates the unsupported transitions by assigning them a constant value. When $|\mathcal{A}| < |\mathcal{Y}|$, both training and decoding times are remarkably reduced by this approach.

We first define the notation for our algorithm. Let $\mathcal{A}_i$ be the active set and $\mathcal{A}_i^c$ be the inactive set of output label i where $\mathcal{Y}_i = \mathcal{A}_i \cup \mathcal{A}_i^c$. We define $\mathcal{A}_i$ as:

$$\mathcal{A}_i = \{j | \delta(y_t = i, y_{t-1} = j) > \epsilon\} \tag{5}$$

where δ is a criterion function of transitions $(y_{t-1} \rightarrow y_t)$ and ϵ is a hyperparameter. For clarity, we define the local factors as:

$$\Psi^1_{t,i} \triangleq \Psi^1_t(y_t = i, \mathbf{x}), \tag{6}$$

$$\Psi^2_{j,i} \triangleq \Psi^2_t(y_{t-1} = j, y_t = i). \tag{7}$$

Note that we can ignore the subscript t at $\Psi^2_t(y_{t-1} = j, y_t = i)$ by defining an HMM-like model, that is, transition matrix $\Psi^2_{j,i}$ is independent of t.

As exact inference, we use the forward-backward procedure to calculate marginals (Sutton and McCallum, 2006). We formally describe here an efficient calculation of α and β recursions for the forward-backward procedure. The forward value $\alpha_t(i)$ is the sum of the unnormalized scores for all partial paths that start at $t = 0$ and converge at $y_t = i$ at time t. The backward value $\beta_t(i)$ similarly defines the sum of unnormalized scores for all partial paths that start at time $t + 1$ with state $y_{t+1} = j$ and continue until the end of the sequences, $t = T + 1$. Then, we decompose the equations of exact α and β recursions as follows:

$$\alpha_t(i) = \Psi^1_{t,i} \left(\sum_{j \in \mathcal{A}_i} \Big(\Psi^2_{j,i} - \omega\Big) \alpha_{t-1}(j) + \omega \right), \tag{8}$$

$$\beta_{t-1}(j) = \sum_{i \in \mathcal{A}_j} \Psi^1_{t,i} \Big(\Psi^2_{j,i} - \omega\Big) \beta_t(i) + \omega \sum_{i \in \mathcal{Y}} \Psi^1_{t,i} \beta_t(i), \tag{9}$$

where ω is a shared transition parameter value for set $\mathcal{A}_i^c$, that is, $\Psi^2_{j,i} = \omega$ if $j \in \mathcal{A}_i^c$. Note that $\sum_i \alpha_t(i) = 1$ (Sutton and McCallum, 2006). Because all unsupported transitions in $\mathcal{A}_i^c$ are calculated simultaneously, the complexities of Eq. (8) and (9) are approximately $\mathcal{O}(T|\mathcal{A}_{avg}||\mathcal{Y}|)$ where $|\mathcal{A}_{avg}|$ is the average number of states in the active set, i.e., $\frac{1}{T}\sum_{t=1}^{T} |\mathcal{A}_i|$. The worst case complexity of our α and β equations is $\mathcal{O}(T|\mathcal{Y}|^2)$.

Similarly, we decompose a γ recursion for the Viterbi algorithm as follows:

$$\gamma_t(i) = \Psi^1_{t,i} \left\{ \max \left(\max_{j \in \mathcal{A}_i} \Psi^2_{j,i} \gamma_{t-1}(j), \max_{j \in \mathcal{Y}} \omega \gamma_{t-1}(j) \right) \right\}, \tag{10}$$

where $\gamma_t(i)$ is the sum of unnormalized scores for the best-scored partial path that starts at time $t = 0$ and converges at $y_t = i$ at time t. Because ω is constant, $\max_{j \in \mathcal{Y}} \gamma_{t-1}(j)$ can be pre-calculated at time $t - 1$. By analogy with Eq. (8) and (9), the complexity is approximately $\mathcal{O}(T|\mathcal{A}_{avg}||\mathcal{Y}|)$.

3.2 Setting δ and ω

To implement our inference algorithm, we need a method of choosing appropriate values for the setting function δ of the active set and for the constant value ω of the inactive set. These two problems are closely related. The size of the active set affects both the complexity of inference algorithm and the quality of the model. Therefore, our goal for selecting δ and ω is to make a plausible assumption that does not sacrifice much accuracy but speeds up when applying large state tasks. We describe four variant special case algorithms.

Method 1: We set $\delta(i, j) = Z(L)$ and $\omega = 0$ where L is a beam set, $L = \{l_1, l_2, \ldots, l_m\}$ and the sub-partition function $Z(L)$ is approximated by $Z(L) \approx \alpha_{t-1}(j)$. In this method, all sub-marginals in the inactive set are totally excluded from calculation of the current marginal. α and β in the inactive sets are set to 0 by default. Therefore, at each time step t the algorithm prunes all states i in which $\alpha_t(i) < \epsilon$. It also generates a subset L of output labels that will be exploited in next time step $t + 1$.[1] This method has been derived theoretically from the process of selecting a compressed marginal distribution within a fixed KL divergence of the true marginal (Pal et al., 2006). This method most closely resembles SFB algorithm; hence we refer an alternative of SFB.

Method 2: We define $\delta(i, j) = |\Psi^2_{j,i} - 1|$ and $\omega = 1$. In practice, unsupported transition features are not parameterized[2]; this means that $\lambda_k = 0$ and $\Psi^2_{j,i} = 1$ if $j \in \mathcal{A}_i^c$. Thus, this method estimates nearly-exact

[1] In practice, dynamically selecting L increases the number of computations, and this is the main disadvantage of Method 1. However, in inactive sets $\alpha_{t-1}(j) = 0$ by default; hence, we need not calculate $\beta_{t-1}(j)$. Therefore, it counterbalances the extra computations in β recursion.

[2] This is a common practice in implementation of input and output joint feature functions for large-scale problems. This scheme uses only supported features that are used at least once in the training examples. We call it the sparse model. While a complete and dense feature model may per-

CRFs if the hyperparameter is $\epsilon = 0$; hence this criterion does not change the parameter. Although this method is simple, it is sufficiently efficient for training and decoding CRFs in real data.

Method 3: We define $\delta(i,j) = E_{\tilde{p}}\langle\phi_k^2(i,j)\rangle$ where $E_{\tilde{p}}\langle z\rangle$ is an empirical count of event z in training data. We also assign a real value for the inactive set, i.e., $\omega = c \in \mathbb{R}, c \neq 0, 1$. The value c is estimated in the training phase; hence, c is a shared parameter for the inactive set. This method is equivalent to TP (Cohn, 2006). By setting ϵ larger, we can achieve faster inference, a tradeoff exists between efficiency and accuracy.

Method 4: We define the shared parameter as a function of output label y in the inactive set, i.e., $c(y)$. As in Method 3, $c(y)$ is estimated during the training phase. When the problem expects different aspects of unsupported transitions, this method would be better than using only one parameter c for all labels in inactive set.

4 Experiment

We evaluated our method on six large-scale natural language data sets (Table 1): Penn Treebank[3] for part-of-speech tagging (PTB), phrase chunking data[4] (CoNLL00), named entity recognition data[5] (CoNLL03), grapheme-to-phoneme conversion data[6] (NetTalk), spoken language understanding data (Communicator) (Jeong and Lee, 2006), and fine-grained named entity recognition data (Encyclopedia) (Lee et al., 2007). The active set is sufficiently small in Communicator and Encyclopedia despite their large numbers of output labels. In all data sets, we selected the current word, ± 2 context words, bigrams, trigrams, and prefix and suffix features as basic feature templates. A template of part-of-speech tag features was added for CoNLL00, CoNLL03, and Encyclopedia. In particular, all tasks except PTB and NetTalk require assigning a label to a phrase rather than to a word; hence, we used standard "BIO" encoding. We used un-normalized log-likelihood, accuracy and training/decoding times as our evaluation measures. We did not use cross validation and development set for tuning the parameter because our goal is to evaluate the efficiency of inference algorithms. Moreover, using the previous state-of-the-art features we expect the achievement of better accuracy.

All our models were trained until parameter estimation converged with a Gaussian prior variance of 4. During training, a pseudo-likelihood parameter estimation (Sutton and McCallum, 2006) was used as an initial weight (estimated in 30 iterations). We used complete and dense input/output joint features for dense model (Dense), and only supported features that are used at least once in the training examples for sparse model (Sparse). All of our model variants were based on Sparse model. For the hyper parameter ϵ, we empirically selected 0.001 for Method 1 (this preserves 99% of probability density), 0 for Method 2, and 4 for Methods 3 and 4. Note that ϵ for Methods 2, 3, and 4 indicates an empirical count of features in training set. All experiments were implemented in C++ and executed in Windows 2003 with XEON 2.33 GHz Quad-Core processor and 8.0 Gbyte of main memory.

Table 1: Data sets: number of sentences in the training (#Train) and the test data sets (#Test), and number of output labels (#Label). $|\mathcal{A}_{avg}^{\omega=1}|$ denotes the average number of active set when $\omega = 1$, i.e., the supported transitions that are used at least once in the training set.

Set	#Train	#Test	#Label	$\lvert\mathcal{A}_{avg}^{\omega=1}\rvert$
PTB	38,219	5462	45	30.01
CoNLL00	8,936	2,012	22	6.59
CoNLL03	14,987	3,684	8	4.13
NetTalk	18,008	2,000	51	22.18
Communicator	13,111	1,193	120	3.67
Encyclopedia	25,348	6,336	279	3.27

We first show that our method is efficient for learning CRFs (Figure 1). In all learning curves, Dense generally has a higher training log-likelihood than Sparse. For PTB and Encyclopedia, results for Dense are not available because training in a single machine failed due to out-of-memory errors. For both Dense and Sparse, we executed the exact inference method. Our proposed method (Method 1$\sim$4) performs faster than Sparse. In most results, Method 1 was the fastest, because it was terminated after fewer iterations. However, Method 1 sometimes failed to converge, for example, in Encyclopedia. Similarly, Method 3 and 4 could not find the optimal solution in the NetTalk data set. Method 2 showed stable results.

Second, we evaluated the accuracy and decoding time of our methods (Table 2). Most results obtained using our method were as accurate as those of Dense and Sparse. However, some results of Method 1, 3, and 4 were significantly inferior to those of Dense and Sparse for one of two reasons: 1) parameter estimation failed (NetTalk and Encyclopedia), or 2) approximate inference caused search errors (CoNLL00 and Communicator). The improvements of decoding time on Communicator and Encyclopedia were remarkable.

Finally, we compared our method with two open-source implementations of CRFs: MALLET[7] and CRF++[8]. MALLET can support the Sparse model, and the CRF++ toolkit implements only the Dense model. We compared them with Method 2 on the Communicator data set. In the accuracy measure, the results were 91.56 (MALLET), 91.87 (CRF++), and 91.92 (ours). Our method performs 5$\sim$50 times faster for training (1,774 s for MALLET, 18,134 s for CRF++,

form better, the sparse model performs well in practice without significant loss of accuracy (Sha and Pereira, 2003).

[3]Penn Treebank3: Catalog No. LDC99T42

[4]http://www.cnts.ua.ac.be/conll2000/chunking/

[5]http://www.cnts.ua.ac.be/conll2003/ner/

[6]http://archive.ics.uci.edu/ml/

[7]Ver. 2.0 RC3, http://mallet.cs.umass.edu/

[8]Ver. 0.51, http://crfpp.sourceforge.net/

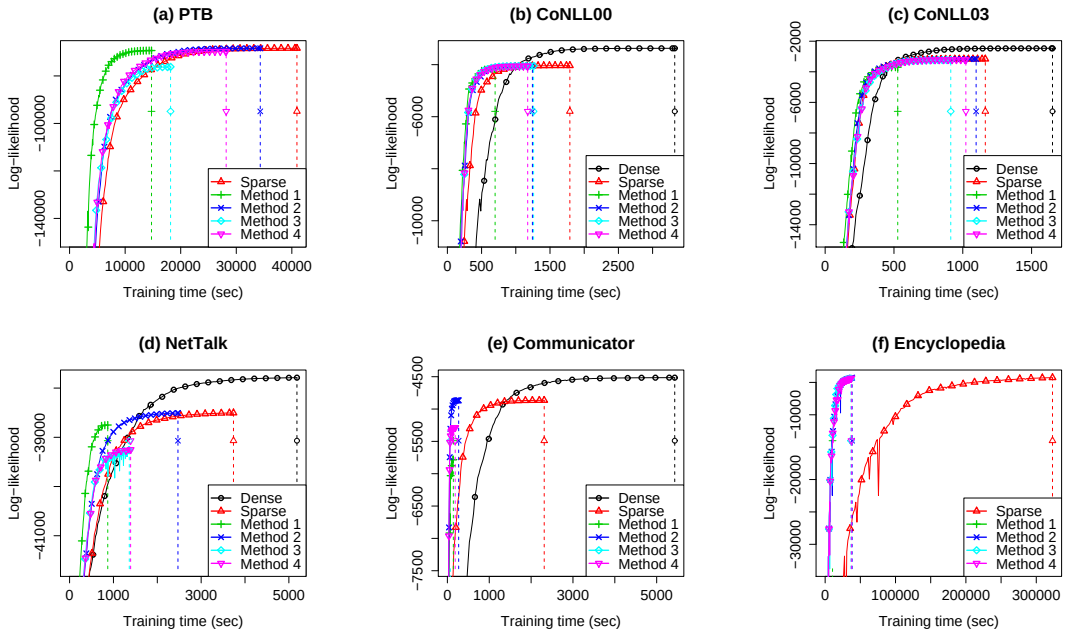


Figure 1: Result of training linear-chain CRFs: Un-normalized training log-likelihood and training times are compared. Dashed lines denote the termination of training step.

Table 2: Decoding result; columns are percent accuracy (Acc), and decoding time in milliseconds (Time) measured per testing example. '*' indicates that the result is significantly different from the Sparse model. N/A indicates failure due to out-of-memory error.

Method	PTB		CoNLL00		CoNLL03		NetTalk		Communicator		Encyclopedia	
	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time
Dense	N/A	N/A	96.1	0.89	95.8	0.26	88.4	0.49	91.6	0.94	N/A	N/A
Sparse	96.6	1.12	95.9	0.62	95.9	0.21	88.4	0.44	91.9	0.83	93.6	34.75
Method 1	96.8	0.74	95.9	0.55	*94.0	0.24	*88.3	0.34	91.7	0.73	*69.2	15.77
Method 2	96.6	0.92	*95.7	0.52	95.9	0.21	*87.4	0.32	91.9	0.30	93.6	4.99
Method 3	96.5	0.84	*94.2	0.51	95.9	0.24	*78.2	0.29	*86.7	0.30	93.7	6.14
Method 4	96.6	0.85	*92.1	0.51	95.9	0.24	*77.9	0.30	91.9	0.29	93.3	4.88

and 368 s for ours) and 7∼12 times faster for decoding (2.881 ms for MALLET, 5.028 ms for CRF++, and 0.418 ms for ours). This result demonstrates that learning and decoding CRFs for large-scale natural language problems can be efficiently solved using our method.

5 Conclusion

We have demonstrated empirically that our efficient inference method can function successfully, allowing for a significant speedup of computation. Our method links two previous algorithms, the SFB and the TP. We have also showed that a simple and robust variant method (Method 2) is effective in large-scale problems.[9] The empirical results show a significant improvement in the training and decoding speeds especially when the problem has a large state space of output labels. Future work will consider applications to other large-scale problems, and more-general graph topologies.

[9]Code used in this work is available at http://argmax.sourceforge.net/.

References

T. Cohn. 2006. Efficient inference in large conditional random fields. In *Proc. ECML*, pages 606–613.

M. Jeong and G. G. Lee. 2006. Exploiting non-local features for spoken language understanding. In *Proc. of COLING/ACL*, pages 412–419, Sydney, Australia, July.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289.

C. Lee, Y. Hwang, and M. Jang. 2007. Fine-grained named entity recognition and relation extraction for question answering. In *Proc. SIGIR Poster*, pages 799–800.

C. Pal, C. Sutton, and A. McCallum. 2006. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *Proc. ICASSP*.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proc. of NAACL/HLT*, pages 134–141.

C. Sutton and A. McCallum. 2006. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.

Iterative Scaling and Coordinate Descent Methods for Maximum Entropy

Fang-Lan Huang, Cho-Jui Hsieh, Kai-Wei Chang, and Chih-Jen Lin
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan
{d93011,b92085,b92084,cjlin}@csie.ntu.edu.tw

Abstract

Maximum entropy (Maxent) is useful in many areas. Iterative scaling (IS) methods are one of the most popular approaches to solve Maxent. With many variants of IS methods, it is difficult to understand them and see the differences. In this paper, we create a general and unified framework for IS methods. This framework also connects IS and coordinate descent (CD) methods. Besides, we develop a CD method for Maxent. Results show that it is faster than existing iterative scaling methods[1].

1 Introduction

Maximum entropy (Maxent) is widely used in many areas such as natural language processing (NLP) and document classification. Maxent models the conditional probability as:

$$P_{\boldsymbol{w}}(y|x) \equiv S_{\boldsymbol{w}}(x,y)/T_{\boldsymbol{w}}(x), \quad (1)$$

$S_{\boldsymbol{w}}(x,y) \equiv e^{\sum_t w_t f_t(x,y)}$, $T_{\boldsymbol{w}}(x) \equiv \sum_y S_{\boldsymbol{w}}(x,y)$, where x indicates a context, y is the label of the context, and $\boldsymbol{w} \in R^n$ is the weight vector. A function $f_t(x,y)$ denotes the t-th feature extracted from the context x and the label y.

Given an empirical probability distribution $\tilde{P}(x,y)$ obtained from training samples, Maxent minimizes the following negative log-likelihood:

$$\begin{aligned}\min_{\boldsymbol{w}} \; & -\sum_{x,y} \tilde{P}(x,y) \log P_{\boldsymbol{w}}(y|x) \\ = & \sum_x \tilde{P}(x) \log T_{\boldsymbol{w}}(x) - \sum_t w_t \tilde{P}(f_t),\end{aligned} \quad (2)$$

where $\tilde{P}(x) = \sum_y \tilde{P}(x,y)$ is the marginal probability of x, and $\tilde{P}(f_t) = \sum_{x,y} \tilde{P}(x,y) f_t(x,y)$ is the expected value of $f_t(x,y)$. To avoid overfitting the training samples, some add a regularization term and solve:

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) \equiv \sum_x \tilde{P}(x) \log T_{\boldsymbol{w}}(x) - \sum_t w_t \tilde{P}(f_t) + \frac{\sum_t w_t^2}{2\sigma^2}, \quad (3)$$

where σ is a regularization parameter. We focus on (3) instead of (2) because (3) is strictly convex.

Iterative scaling (IS) methods are popular in training Maxent models. They all share the same property of *solving a one-variable sub-problem at a time.* Existing IS methods include generalized iterative scaling (GIS) by Darroch and Ratcliff (1972), improved iterative scaling (IIS) by Della Pietra et al. (1997), and sequential conditional generalized iterative scaling (SCGIS) by Goodman (2002). In optimization, coordinate descent (CD) is a popular method which also *solves a one-variable sub-problem at a time.* With these many IS and CD methods, it is uneasy to see their differences. In Section 2, we propose a unified framework to describe IS and CD methods from an optimization viewpoint. Using this framework, we design a fast CD approach for Maxent in Section 3. In Section 4, we compare the proposed CD method with IS and LBFGS methods. Results show that the CD method is more efficient.

Notation n is the number of features. The total number of nonzeros in samples and the average number of nonzeros per feature are respectively

$$\#\text{nz} \equiv \sum_{x,y} \sum_{t:f_t(x,y)\neq 0} 1 \quad \text{and} \quad \bar{l} \equiv \#\text{nz}/n.$$

2 A Framework for IS Methods

2.1 The Framework

The one-variable sub-problem of IS methods is related to the function reduction $L(\boldsymbol{w}+z\boldsymbol{e}_t) - L(\boldsymbol{w})$, where $\boldsymbol{e}_t = [0,\ldots,0,1,0,\ldots,0]^T$. IS methods differ in how they approximate the function reduction. They can also be categorized according to whether $\boldsymbol{w}$'s components are sequentially or parallely updated. In this section, we create a framework in Figure 1 for these methods.

Sequential update For a sequential-update algorithm, once a one-variable sub-problem is solved, the corresponding element in $\boldsymbol{w}$ is updated. The new $\boldsymbol{w}$ is then used to construct the next sub-problem. The procedure is sketched in

[1] A complete version of this work is at http://www.csie.ntu.edu.tw/~cjlin/papers/maxent_journal.pdf.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 285–288,
Suntec, Singapore, 4 August 2009.

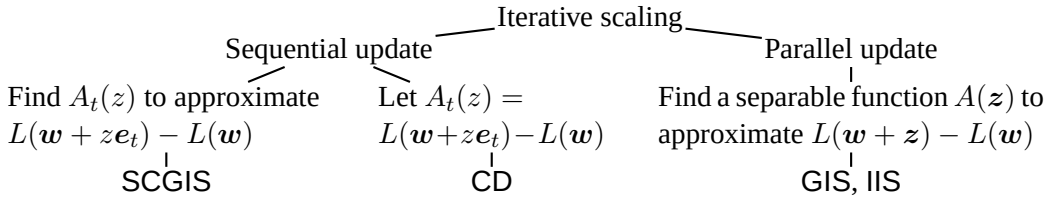


Figure 1: An illustration of various iterative scaling methods.

Algorithm 1 A sequential-update IS method

While $\boldsymbol{w}$ is not optimal

For $t = 1, \ldots, n$

1. Find an approximate function $A_t(z)$ satisfying (4).
2. Approximately $\min_z A_t(z)$ to get $\bar{z}_t$.
3. $w_t \leftarrow w_t + \bar{z}_t$.

Algorithm 1. If the t-th component is selected for update, a sequential IS method solves the following one-variable sub-problem:

$$\min_z \ A_t(z),$$

where $A_t(z)$ bounds the function difference:

$$A_t(z) \geq L(\boldsymbol{w}+z\boldsymbol{e}_t) - L(\boldsymbol{w}) = \sum_x \tilde{P}(x) \log \frac{T_{\boldsymbol{w}+z\boldsymbol{e}_t}(x)}{T_{\boldsymbol{w}}(x)} + Q_t(z) \quad (4)$$

$$\text{and} \quad Q_t(z) \equiv \frac{2w_t z + z^2}{2\sigma^2} - z\tilde{P}(f_t). \quad (5)$$

An approximate function $A_t(z)$ satisfying (4) does not ensure that the function value is strictly decreasing. That is, the new function value $L(\boldsymbol{w}+z\boldsymbol{e}_t)$ may be only the same as $L(\boldsymbol{w})$. Therefore, we can impose an additional condition

$$A_t(0) = 0 \quad (6)$$

on the approximate function $A_t(z)$. If $A'_t(0) \neq 0$ and assume $\bar{z}_t \equiv \arg\min_z A_t(z)$ exists, with the condition $A_t(0) = 0$, we have $A_t(\bar{z}_t) < 0$. This inequality and (4) then imply $L(\boldsymbol{w}+\bar{z}_t\boldsymbol{e}_t) < L(\boldsymbol{w})$. If $A'_t(0) = \nabla_t L(\boldsymbol{w}) = 0$, the convexity of $L(\boldsymbol{w})$ implies that we cannot decrease the function value by modifying w_t. Then we should move on to modify other components of $\boldsymbol{w}$.

A CD method can be viewed as a sequential IS method. It solves the following sub-problem:

$$\min_z \ A_t^{\mathsf{CD}}(z) = L(\boldsymbol{w}+z\boldsymbol{e}_t) - L(\boldsymbol{w})$$

without any approximation. Existing IS methods consider approximations as $A_t(z)$ may be simpler for minimization.

Parallel update A parallel IS method simultaneously constructs n independent one-variable sub-problems. After (approximately) solving all of them, the whole vector $\boldsymbol{w}$ is updated. Algorithm 2 gives the procedure. The differentiable function $A(\boldsymbol{z}),\ \boldsymbol{z} \in R^n$, is an approximation of $L(\boldsymbol{w}+\boldsymbol{z}) - L(\boldsymbol{w})$ satisfying

$$A(\boldsymbol{z}) \geq L(\boldsymbol{w}+\boldsymbol{z}) - L(\boldsymbol{w}),\ \ A(\boldsymbol{0}) = 0, \ \text{and} \ A(\boldsymbol{z}) = \sum_t A_t(z_t). \quad (7)$$

Similar to (4) and (6), the first two conditions ensure that the function value is strictly decreasing. The last condition shows that $A(\boldsymbol{z})$ is separable, so $\min_{\boldsymbol{z}} A(\boldsymbol{z}) = \sum_t \min_{z_t} A_t(z_t)$. That is, we can minimize $A_t(z_t), \forall t$ simultaneously, and then update $w_t\ \forall t$ together. A parallel-update method possesses nice implementation properties. However, since it less aggressively updates $\boldsymbol{w}$, it usually converges slower. If $A(\boldsymbol{z})$ satisfies (7), taking $\boldsymbol{z} = z_t\boldsymbol{e}_t$ implies that (4) and (6) hold for any $A_t(z_t)$. A parallel method could thus be transformed to a sequential method using the same approximate function, but not vice versa.

Algorithm 2 A parallel-update IS method

While $\boldsymbol{w}$ is not optimal

1. Find approximate functions $A_t(z_t)\ \forall t$ satisfying (7).
2. For $t = 1, \ldots, n$
   Approximately $\min_{z_t} A_t(z_t)$ to get $\bar{z}_t$.
3. For $t = 1, \ldots, n$
   $w_t \leftarrow w_t + \bar{z}_t$.

2.2 Existing Iterative Scaling Methods

We introduce GIS, IIS and SCGIS via the proposed framework. GIS and IIS use a parallel update, but SCGIS is sequential. Their approximate functions aim to bound the function reduction

$$L(\boldsymbol{w}+\boldsymbol{z}) - L(\boldsymbol{w}) = \sum_x \tilde{P}(x) \log \frac{T_{\boldsymbol{w}+\boldsymbol{z}}(x)}{T_{\boldsymbol{w}}(x)} + \sum_t Q_t(z_t), \quad (8)$$

where $T_{\boldsymbol{w}}(x)$ and $Q_t(z_t)$ are defined in (1) and (5), respectively. Then GIS, IIS and SCGIS use similar inequalities to get approximate functions. They apply $\log \alpha \leq \alpha - 1\ \forall \alpha > 0$ to get

$$(8) \leq \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) (e^{\sum_t z_t f_t(x,y)} - 1) + \sum_t Q_t(z_t). \quad (9)$$

GIS defines

$$f^{\#} \equiv \max_{x,y} f^{\#}(x,y), \ \ f^{\#}(x,y) \equiv \sum_t f_t(x,y),$$

and adds a feature $f_{n+1}(x,y) \equiv f^{\#} - f^{\#}(x,y)$ with $z_{n+1} = 0$. Assuming $f_t(x,y) \geq 0,\ \forall t, x, y$, and using Jensen's inequality

$$e^{\sum_{t=1}^{n+1} \frac{f_t(x,y)}{f^{\#}} (z_t f^{\#})} \leq \sum_{t=1}^{n+1} \frac{f_t(x,y)}{f^{\#}} e^{z_t f^{\#}} \ \text{and}$$

$$e^{\sum_t z_t f_t(x,y)} \leq \sum_t \frac{f_t(x,y)}{f^{\#}} e^{z_t f^{\#}} + \frac{f_{n+1}(x,y)}{f^{\#}}, \quad (10)$$

we obtain n independent one-variable functions:

$$A_t^{\mathsf{GIS}}(z_t) = \frac{e^{z_t f^{\#}} - 1}{f^{\#}} \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) + Q_t(z_t).$$

IIS applies Jensen's inequality

$$e^{\sum_t \frac{f_t(x,y)}{f^\#(x,y)}(z_t f^\#(x,y))} \le \sum_t \frac{f_t(x,y)}{f^\#(x,y)} e^{z_t f^\#(x,y)}$$

on (9) to get the approximate function

$$A_t^{\mathsf{IIS}}(z_t) = \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) \frac{e^{z_t f^\#(x,y)} - 1}{f^\#(x,y)} + Q_t(z_t).$$

SCGIS is a sequential-update method. It replaces $f^\#$ in GIS with $f_t^\# \equiv \max_{x,y} f_t(x,y)$. Using $z_t \boldsymbol{e}_t$ as $\boldsymbol{z}$ in (8), a derivation similar to (10) gives

$$e^{z_t f_t(x,y)} \le \frac{f_t(x,y)}{f_t^\#} e^{z_t f_t^\#} + \frac{f_t^\# - f_t(x,y)}{f_t^\#}.$$

The approximate function of SCGIS is

$$A_t^{\mathsf{SCGIS}}(z_t) = \frac{e^{z_t f_t^\#} - 1}{f_t^\#} \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) + Q_t(z_t).$$

We prove the linear convergence of existing IS methods (proof omitted):

Theorem 1 *Assume each sub-problem $A_t^s(z_t)$ is exactly minimized, where s is* IIS, GIS, SCGIS, *or* CD. *The sequence $\{\boldsymbol{w}^k\}$ generated by any of these four methods linearly converges. That is, there is a constant $\mu \in (0,1)$ such that $L(\boldsymbol{w}^{k+1}) - L(\boldsymbol{w}^*) \le (1-\mu)(L(\boldsymbol{w}^k) - L(\boldsymbol{w}^*)), \forall k$, where $\boldsymbol{w}^*$ is the global optimum of* (3).

2.3 Solving one-variable sub-problems

Without the regularization term, by $A_t'(z_t) = 0$, GIS and SCGIS both have a simple closed-form solution of the sub-problem. With the regularization term, the sub-problems no longer have a closed-form solution. We discuss the cost of solving sub-problems by the Newton method, which iteratively updates z_t by

$$z_t \leftarrow z_t - A_t^{s\prime}(z_t) / A_t^{s\prime\prime}(z_t). \quad (11)$$

Here s indicates an IS or a CD method.

Below we check the calculation of $A_t^{s\prime}(z_t)$ as the cost of $A_t^{s\prime\prime}(z_t)$ is similar. We have

$$A_t^{s\prime}(z_t) = \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y) e^{z_t f^s(x,y)} + Q_t'(z_t) \quad (12)$$

where

$$f^s(x,y) \equiv \begin{cases} f^\# \text{ if } s \text{ is GIS}, \\ f_t^\# \text{ if } s \text{ is SCGIS}, \\ f^\#(x,y) \text{ if } s \text{ is IIS}. \end{cases}$$

For CD,

$$A_t^{\mathsf{CD}\prime}(z_t) = Q_t'(z_t) + \sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x) f_t(x,y). \quad (13)$$

The main cost is on calculating $P_{\boldsymbol{w}}(y|x)\ \forall x, y$, whenever $\boldsymbol{w}$ is updated. Parallel-update approaches calculate $P_{\boldsymbol{w}}(y|x)$ once every n sub-problems, but sequential-update methods evaluates $P_{\boldsymbol{w}}(y|x)$ after every sub-problem. Consider the situation of updating $\boldsymbol{w}$ to $\boldsymbol{w} + z_t\boldsymbol{e}_t$. By (1), obtaining $P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x)\ \forall x, y$ requires expensive $O(\#\text{nz})$ operations to evaluate $S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y)$ and $T_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x)\ \forall x, y$. A trick to trade memory for time is to store all $S_{\boldsymbol{w}}(x,y)$ and $T_{\boldsymbol{w}}(x)$,

$$S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y) = S_{\boldsymbol{w}}(x,y) e^{z_t f_t(x,y)},$$

$$T_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x) = T_{\boldsymbol{w}}(x) + \sum_y S_{\boldsymbol{w}}(x,y)(e^{z_t f_t(x,y)} - 1).$$

Since $S_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(x,y) = S_{\boldsymbol{w}}(x,y)$ if $f_t(x,y) = 0$, this procedure reduces the the $O(\#\text{nz})$ operations to $O(\#\text{nz}/n) = O(\bar{l})$. However, it needs extra spaces to store all $S_{\boldsymbol{w}}(x,y)$ and $T_{\boldsymbol{w}}(x)$. This trick for updating $P_{\boldsymbol{w}}(y|x)$ has been used in SCGIS (Goodman, 2002). Thus, the first Newton iteration of all methods discussed here takes $O(\bar{l})$ operations. For each subsequent Newton iteration, CD needs $O(\bar{l})$ as it calculates $P_{\boldsymbol{w}+z_t\boldsymbol{e}_t}(y|x)$ whenever z_t is changed. For GIS and SCGIS, if $\sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y)$ is stored at the first Newton iteration, then (12) can be done in $O(1)$ time. For IIS, because $f^\#(x,y)$ of (12) depends on x and y, we cannot store $\sum_{x,y} \tilde{P}(x) P_{\boldsymbol{w}}(y|x) f_t(x,y)$ as in GIS and SCGIS. Hence each Newton direction needs $O(\bar{l})$. We summarize the cost for solving sub-problems in Table 1.

Table 1: Time for minimizing $A_t(z_t)$ by the Newton method

	CD	GIS	SCGIS	IIS
1st Newton direction	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$	$O(\bar{l})$
Each subsequent Newton direction	$O(\bar{l})$	$O(1)$	$O(1)$	$O(\bar{l})$

3 Comparison and a New CD Method

3.1 Comparison of IS/CD methods

From the above discussion, an IS or a CD method falls into a place between two extreme designs:

$A_{t(z_t)}$ a loose bound $\quad\leftrightarrow\quad$ $A_{t(z_t)}$ a tight bound
Easy to minimize $A_{t(z_t)}$ $\quad\leftrightarrow\quad$ Hard to minimize $A_{t(z_t)}$

There is a tradeoff between the tightness to bound the function difference and the hardness to solve the sub-problem. To check how IS and CD methods fit into this explanation, we obtain relationships of their approximate functions:

$$A_t^{\mathsf{CD}}(z_t) \le A_t^{\mathsf{SCGIS}}(z_t) \le A_t^{\mathsf{GIS}}(z_t), \quad A_t^{\mathsf{CD}}(z_t) \le A_t^{\mathsf{IIS}}(z_t) \le A_t^{\mathsf{GIS}}(z_t) \quad \forall\ z_t. \quad (14)$$

The derivation is omitted. From (14), CD considers more accurate sub-problems than SCGIS and GIS. However, to solve each sub-problem, from Table 1, CD's each Newton step takes more time. The same situation occurs in comparing IIS and GIS. Therefore, while a tight $A_t(z_t)$ can

give faster convergence by handling fewer sub-problems, the total time may not be less due to the higher cost of each sub-problem.

3.2 A Fast CD Method

We develop a CD method which is cheaper in solving each sub-problem but still enjoys fast final convergence. This method is modified from Chang et al. (2008), a CD approach for linear SVM. We approximately minimize $A_t^{\mathsf{CD}}(z)$ by applying only one Newton iteration. The Newton direction at $z = 0$ is now

$$d = -A_t^{\mathsf{CD}\prime}(0)/A_t^{\mathsf{CD}\prime\prime}(0). \quad (15)$$

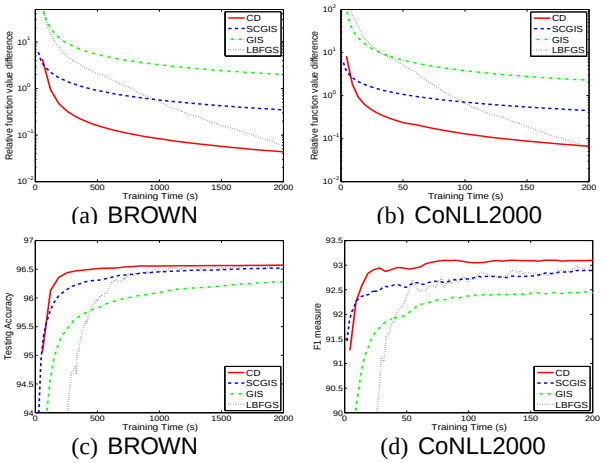
As taking the full Newton direction may not decrease the function value, we need a line search procedure to find $\lambda \geq 0$ such that $z = \lambda d$ satisfies the following sufficient decrease condition:

$$A_t^{\mathsf{CD}}(z) - A_t^{\mathsf{CD}}(0) = A_t^{\mathsf{CD}}(z) \leq \gamma z A_t^{\mathsf{CD}\prime}(0), \quad (16)$$

where γ is a constant in $(0, 1/2)$. A simple way to find λ is by sequentially checking $\lambda = 1, \beta, \beta^2, \ldots$, where $\beta \in (0, 1)$. The line search procedure is guaranteed to stop (proof omitted). We can further prove that near the optimum two results hold: First, the Newton direction (15) satisfies the sufficient decrease condition (16) with $\lambda = 1$. Then the cost for each sub-problem is $O(\bar{l})$, similar to that for exactly solving sub-problems of GIS or SCGIS. This result is important as otherwise each trial of $z = \lambda d$ expensively costs $O(\bar{l})$ for calculating $A_t^{\mathsf{CD}}(z)$. Second, taking one Newton direction of the tighter $A_t^{\mathsf{CD}}(z_t)$ reduces the function $L(\boldsymbol{w})$ more rapidly than exactly minimizing a loose $A_t(z_t)$ of GIS, IIS or SCGIS. These two results show that the new CD method improves upon the traditional CD by approximately solving sub-problems, while still maintains fast convergence.

4 Experiments

We apply Maxent models to part of speech (POS) tagging for BROWN corpus (`http://www.nltk.org`) and chunking tasks for CoNLL2000 (`http://www.cnts.ua.ac.be/conll2000/chunking`). We randomly split the BROWN corpus to 4/5 training and 1/5 testing. Our implementation is built upon OpenNLP (`http://maxent.sourceforge.net`). We implement CD (the new one in Section 3.2), GIS, SCGIS, and LBFGS for comparisons. We include LBFGS as Malouf (2002) reported that it is better than other approaches including GIS and IIS. We use $\sigma^2 = 10$, and set $\beta = 0.5$ and $\gamma = 0.001$ in (16).



(a) BROWN (b) CoNLL2000

(c) BROWN (d) CoNLL2000

Figure 2: First row: time versus the relative function value difference (17). Second row: time versus testing accuracy/F1. Time is in seconds.

We begin at checking time versus the relative difference of the function value to the optimum:

$$L(\boldsymbol{w}) - L(\boldsymbol{w}^*)/L(\boldsymbol{w}^*). \quad (17)$$

Results are in the first row of Figure 2. We check in the second row of Figure 2 about testing accuracy/F1 versus training time. Among the three IS/CD methods compared, the new CD approach is the fastest. SCGIS comes the second, while GIS is the last. This result is consistent with the tightness of their approximation functions; see (14). LBFGS has fast final convergence, but it does not perform well in the beginning.

5 Conclusions

In summary, we create a general framework for explaining IS methods. Based on this framework, we develop a new CD method for Maxent. It is more efficient than existing IS methods.

References

K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. 2008. Coordinate descent method for large-scale L2-loss linear SVM. *JMLR*, 9:1369–1398.

John N. Darroch and Douglas Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Ann. Math. Statist.*, 43(5):1470–1480.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE PAMI*, 19(4):380–393.

Joshua Goodman. 2002. Sequential conditional generalized iterative scaling. In *ACL*, pages 9–16.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *CONLL*.

Automatic Cost Estimation for Tree Edit Distance Using Particle Swarm Optimization

Yashar Mehdad
University of Trento and FBK - Irst
Trento, Italy
mehdad@fbk.eu

Abstract

Recently, there is a growing interest in working with tree-structured data in different applications and domains such as computational biology and natural language processing. Moreover, many applications in computational linguistics require the computation of similarities over pair of syntactic or semantic trees. In this context, Tree Edit Distance (TED) has been widely used for many years. However, one of the main constraints of this method is to tune the cost of edit operations, which makes it difficult or sometimes very challenging in dealing with complex problems. In this paper, we propose an original method to estimate and optimize the operation costs in TED, applying the Particle Swarm Optimization algorithm. Our experiments on Recognizing Textual Entailment show the success of this method in automatic estimation, rather than manual assignment of edit costs.

1 Introduction

Among many tree-based algorithms, Tree Edit Distance (TED) has offered many solutions for various NLP applications such as information retrieval, information extraction, similarity estimation and textual entailment. Tree edit distance is defined as the minimum costly set of basic operations transforming one tree to another. In common, TED approaches use an initial fixed cost for each operation.

Generally, the initial assigned cost to each edit operation depends on the nature of nodes, applications and dataset. For example the probability of deleting a function word from a string is not the same as deleting a symbol in RNA structure. According to this fact, tree comparison may be affected by application and dataset. A solution to this problem is assigning the cost to each edit operation empirically or based on the expert knowledge and recommendation. These methods emerge a critical problem when the domain, field or application is new and the level of expertise and empirical knowledge is very limited.

Other approaches towards this problem tried to learn a generative or discriminative probabilistic model (Bernard et al., 2008) from the data. One of the drawbacks of those approaches is that the cost values of edit operations are hidden behind the probabilistic model. Additionally, the cost can not be weighted or varied according to the tree context and node location.

In order to overcome these drawbacks, we are proposing a stochastic method based on Particle Swarm Optimization (PSO) to estimate the cost of each edit operation based on the user defined application and dataset. A further advantage of the method, besides automatic learning of the operation costs, is to investigate the cost values in order to better understand how TED approaches the application and data in different domains.

As for the experiments, we learn a model for recognizing textual entailment, based on TED, where the input is a pair of strings represented as syntactic dependency trees. Our results illustrate that optimizing the cost of each operation can dramatically affect the accuracy and achieve a better model for recognizing textual entailment.

2 Tree Edit Distance

Tree edit distance measure is a similarity metric for rooted ordered trees. Assuming that we have two rooted and ordered trees, it means that one node in each tree is assigned as a root and the children of each node are ordered. The edit operations on the nodes a and b between trees are defined as: *Insertion* ($\lambda \rightarrow a$), *Deletion* ($a \rightarrow \lambda$) and *Substitution* ($a \rightarrow b$). Each edit operation has

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 289–292,
Suntec, Singapore, 4 August 2009.

an associated cost (denoted as $\gamma(a \rightarrow b)$). An edit script on two trees is a sequence of edit operations changing a tree to another. Consequently, the cost of an edit script is the sum of the costs of its edit operations. Based on the main definition of this approach, TED is the cost of minimum cost edit script between two trees (Zhang and Shasha, 1989).

In the classic TED, a cost value is assigned to each operation initially, and the distance is computed based on the initial cost values. Considering that the distance can vary in different domains and datasets, converging to an optimal set of values for operations is almost empirically impossible. In the following sections, we propose a method for estimating the optimum set of values for operation costs in TED algorithm. Our method is built on adapting the PSO optimization approach as a search process to automate the procedure of cost estimation.

3 Particle Swarm Optimization

PSO is a stochastic optimization technique which was introduced recently based on the social behaviour of bird flocking and fish schooling (Eberhart et al., 2001). PSO is one of the population-based search methods which takes advantage of the concept of social sharing of information. In this algorithm each *particle* can learn from the experience of other particles in the same population (called *swarm*). In other words, each particle in the iterative search process would adjust its flying velocity as well as position not only based on its own acquaintance but also other particles' flying experience in the swarm. This algorithm has found efficient in solving a number of engineering problems. PSO is mainly built on the following equations.

$$X_i = X_i + V_i \tag{1}$$

$$V_i = \omega V_i + c_1 r_1 (X_{bi} - X_i) + c_2 r_2 (X_{gi} - X_i) \tag{2}$$

To be concise, for each particle at each iteration, the position X_i (Equation 1) and velocity V_i (Equation 2) is updated. X_{bi} is the best position of the particle during its past routes and X_{gi} is the best global position over all routes travelled by the particles of the swarm. r_1 and r_2 are random variables drawn from a uniform distribution in the range [0,1], while c_1 and c_2 are two acceleration constants regulating the relative velocities with respect to the best local and global positions. The weight ω is used as a tradeoff between the global and local best positions. It is usually selected slightly less than 1 for better global exploration (Melgani and Bazi, 2008). Position optimally is computed based on the fitness function defined in association with the related problem. Both position and velocity are updated during the iterations until convergence is reached or iterations attain the maximum number defined by the user.

4 Automatic Cost Optimization for TED

In this section we proposed a system for estimating and optimizing the cost of each edit operation for TED. As mentioned earlier, the aim of this system is to find the optimal set of operation costs to: 1) improve the performance of TED in different applications, and 2) provide some information on how different operations in TED approach an application or dataset. In order to obtain this, the system is developed using an optimization framework based on PSO.

4.1 PSO Setup

One of the most important steps in applying PSO is to define a fitness function, which could lead the swarm to the optimized particles based on the application and data. The choice of this function is very crucial since, based on this, PSO evaluates the quality of each candidate particle for driving the solution space to optimization. Moreover, this function should be, possibly, application and data independent, as well as flexible enough to be adapted to the TED based problems. With the intention of accomplishing these goals, we define two main fitness functions as follows:

1) Bhattacharyya Distance: This statistical measure determines the similarity of two discrete probability distributions (Bhattacharyya, 1943). In classification, this method is used to measure the distance between two different classes. Put it differently, maximizing the Bhattacharyya distance would increase the separability of two classes.

2) Accuracy: By maximizing the accuracy obtained from 10 fold cross-validation on the development set, as the fitness function, we estimate the optimized cost of the edit operations.

4.2 Integrating TED with PSO

The procedure to estimate and optimize the cost of edit operations in TED applying the PSO algorithm, is as follows.

a) *Initialization*

1) Generate a random swarm of size n (cost of edit operations).
2) For each position of the particle from the swarm, obtain the fitness function value.
3) Set the best position of each particle with its initial position (X_{bi}).

b) *Search*

4) Detect the best global position (X_{gi}) in the swarm based on maximum value of the fitness function over all explored routes.
5) Update the velocity of each particle (V_i).
6) Update the position of each particle (X_i).
7) For each candidate particle calculate the fitness function.
8) Update the best position of each particle if the current position has a larger value.

c) *Convergence*

9) Run till the maximum number of iteration (in our case set to 10) is reached or start the search process.

5 Experimental Design

Our experiments were conducted on the basis of Recognizing Textual Entailment (RTE) datasets[1]. Textual Entailment can be explained as an association between a coherent text(T) and a language expression, called hypothesis(H). The entailment function for the pair T-H returns the true value when the meaning of H can be inferred from the meaning of T and false otherwise. In another word, Textual Entailment can be defined as human reading comprehension task. One of the approaches to textual entailment problem is based on the distance between T and H.

In this approach, the entailment score for a pair is calculated on the minimal set of edit operations that transform T into H. An entailment relation is assigned to a T-H pair in the case that overall cost of the transformations is below a certain threshold. The threshold, which corresponds to tree edit distace, is empirically estimated over the dataset. This method was implemented by (Kouylekov and Magnini, 2005), based on TED algorithm (Zhang and Shasha, 1989). Each RTE dataset includes its own development and test set, however, RTE-4 was released only as a test set and the data from RTE-1 to RTE-3 were exploited as development set for evaluating RTE-4 data.

In order to deal with TED approach to textual entailment, we used EDITS[2] package (Edit Distance Textual Entailment Suite) (Magnini et al., 2009). In addition, We partially exploit JSwarm-PSO[3] package with some adaptations as an implementation of PSO algorithm. Each pair in the datasets converted to two syntactic dependency trees using Stanford statistical parser[4], developed in the Stanford university NLP group by (Klein and Manning, 2003).

We conducted six different experiments in two sets on each RTE dataset. The costs were estimated on the training set, then we evaluate the estimated costs on the test set. In the first set of experiments, we set a simple cost scheme based on three operations. Implementing this cost scheme, we expect to optimize the cost of each edit operation without considering that the operation costs may vary based on different characteristics of a node, such as size, location or content. The results were obtained using: 1) The random cost assignment, 2) Assigning the cost based on the expertise knowledge and intuition (So called Intuitive), and 3) Automatic estimated and optimized cost for each operation. In the second case, we applied the same cost values which was used in EDITS by its developers (Magnini et al., 2009).

In the second set of experiments, we tried to take advantage of an advanced cost scheme with more fine-grained operations to assign a weight to the edit operations based on the characteristics of the nodes (Magnini et al., 2009). For example if a node is in the list of stop-words, the deletion cost should be different from the cost of deleting a content word. By this intuition, we tried to optimize 9 specialized costs for edit operations (A swarm of size 9). At each experiment, both fitness functions were applied and the best results were chosen for presentation.

[1] http://www.pascal-network.org/Challenges/RTE1-4

[2] http://edits.fbk.eu/

[3] http://jswarm-pso.sourceforge.net/

[4] http://nlp.stanford.edu/software/lex-parser.shtml

Model		Data set RTE4	RTE3	RTE2	RTE1
Simple	Random	49.6	53.62	50.37	50.5
	Intuitive	51.3	59.6	56.5	49.8
	Optimized	56.5	61.62	58	58.12
Adv.	Random	53.60	52.0	54.62	53.5
	Intuitive	57.6	59.37	57.75	55.5
	Optimized	59.5	62.4	59.87	58.62
Baseline		57.19			
RTE-4 Challenge		57.0			

Table 1: Comparison of accuracy on all RTE datasets based on optimized and unoptimized cost schemes.

6 Results

Our results are summarized in Table 1. We show the accuracy gained by a distance-based baseline for textual entailment (Mehdad and Magnini, 2009) in compare with the results achieved by the random, intuitive and optimized cost schemes using EDITS system. For the better comparison, we also present the results of the EDITS system (Cabrio et al., 2008) in RTE-4 challenge using combination of different distances as features for classification (Cabrio et al., 2008).

Table 1 shows that, in all datasets, accuracy improved up to 9% by optimizing the cost of each edit operation. Results prove that, the optimized cost scheme enhances the quality of the system performance even more than the cost scheme used by the experts (Intuitive cost scheme). Furthermore, using the fine-grained and weighted cost scheme for edit operations we could achieve the highest results in accuracy. Moreover, by exploring the estimated optimal cost of each operation, we could find even some linguistics phenomena which exists in the dataset. For instance, in most of the cases, the cost of deletion was estimated zero, which shows that deleting the words from the text does not effect the distance in the entailment pairs. In addition, the optimized model can reflect more consistency and stability (from 58 to 62 in accuracy) than other models, while in unoptimized models the result varies more, on different datasets (from 50 in RTE-1 to 59 in RTE-3).

7 Conclusion

In this paper, we proposed a novel approach for estimating the cost of edit operations in TED. This model has the advantage of being efficient and more transparent than probabilistic approaches as well as having less complexity. The easy implementation of this approach, besides its flexibility, makes it suitable to be applied in real world applications. The experimental results on textual entailment, as one of the challenging problems in NLP, confirm our claim.

Acknowledgments

Besides my special thanks to F. Melgani, B. Magnini and M. Kouylekov for their academic and technical support, I acknowledge the reviewers for their comments. The EDITS system has been supported by the EU-funded project QALL-ME (FP6 IST-033860).

References

M. Bernard, L. Boyer, A. Habrard, and M. Sebban. 2008. Learning probabilistic models of tree edit distance. *Pattern Recogn.*, 41(8):2611–2629.

A. Bhattacharyya. 1943. On a measure of divergence between two statistical populations defined by probability distributions. *Bull. Calcutta Math. Soc.*, 35:99109.

E. Cabrio, M. Kouylekovand, and B. Magnini. 2008. Combining specialized entailment engines for rte-4. In *Proceedings of TAC08, 4th PASCAL Challenges Workshop on Recognising Textual Entailment*.

R. C. Eberhart, Y. Shi, and J. Kennedy. 2001. *Swarm Intelligence*. The Morgan Kaufmann Series in Artificial Intelligence.

D. Klein and C. D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA. MIT Press.

M. Kouylekov and B. Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *PASCAL Challenges on RTE*, pages 17–20.

B. Magnini, M. Kouylekov, and E. Cabrio. 2009. Edits - Edit Distance Textual Entailment Suite User Manual. Available at *http://edits.fbk.eu/*.

Y. Mehdad and B. Magnini. 2009. A word overlap baseline for the recognizing textual entailment task. Available at *http://edits.fbk.eu/*.

F. Melgani and Y. Bazi. 2008. Classification of electrocardiogram signals with support vector machines and particle swarm optimization. *IEEE Transactions on Information Technology in Biomedicine*, 12(5):667–677.

K. Zhang and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.

Markov Random Topic Fields

Hal Daumé III
School of Computing
University of Utah
Salt Lake City, UT 84112
me@hal3.name

Abstract

Most approaches to topic modeling assume an independence between documents that is frequently violated. We present an topic model that makes use of one or more user-specified graphs describing relationships between documents. These graph are encoded in the form of a Markov random field over topics and serve to encourage related documents to have similar topic structures. Experiments on show upwards of a 10% improvement in modeling performance.

1 Introduction

One often wishes to apply topic models to large document collections. In these large collections, we usually have meta-information about how one document relates to another. Perhaps two documents share an author; perhaps one document cites another; perhaps two documents are published in the same journal or conference. We often believe that documents related in such a way should have similar topical structures. We encode this in a probabilistic fashion by imposing an (undirected) Markov random field (MRF) on top of a standard topic model (see Section 3). The edge potentials in the MRF encode the fact that "connected" documents should share similar topic structures, measured by some parameterized distance function. Inference in the resulting model is complicated by the addition of edge potentials in the MRF. We demonstrate that a hybrid Gibbs/Metropolis-Hastings sampler is able to efficiently explore the posterior distribution (see Section 4).

In experiments (Section 5), we explore several variations on our basic model. The first is to explore the importance of being able to tune the strength of the potentials in the MRF as part of the inference procedure. This turns out to be of utmost importance. The second is to study the importance of the form of the distance metric used to specify the edge potentials. Again, this has a significant impact on performance. Finally, we consider the use of multiple graphs for a single model and find that the power of combined graphs also leads to significantly better models.

2 Background

Probabilistic topic models propose that text can be considered as a mixture of words drawn from one or more "topics" (Deerwester et al., 1990; Blei et al., 2003). The model we build on is latent Dirichlet allocation (Blei et al., 2003) (henceforth, LDA). LDA stipulates the following generative model for a document collection:

1. For each document $d = 1 \dots D$:
   (a) Choose a topic mixture $\boldsymbol{\theta}_d \sim \mathcal{D}ir(\boldsymbol{\alpha})$
   (b) For each word in d, $n = 1 \dots N_d$:
      i. Choose a topic $z_{dn} \sim \mathcal{M}ult(\boldsymbol{\theta}_d)$
      ii. Choose a word $w_{dn} \sim \mathcal{M}ult(\boldsymbol{\beta}_{z_{dn}})$

Here, $\boldsymbol{\alpha}$ is a hyperparameter vector of length K, where K is the desired number of topics. Each document has a topic distribution $\boldsymbol{\theta}_d$ over these K topics and each word is associated with precisely one topic (indicated by z_{dn}). Each topic $k = 1 \dots K$ is a unigram distribution over words (aka, a multinomial) parameterized by a vector $\boldsymbol{\beta}_k$. The associated graphical model for LDA is shown in Figure 1. Here, we have added a few additional hyperparameters: we place a $\mathcal{G}am(a, b)$ prior independently on each component of $\boldsymbol{\alpha}$ and a $\mathcal{D}ir(\eta, \dots, \eta)$ prior on each of the $\boldsymbol{\beta}$s.

The joint distribution over all random variables specified by LDA is:

$$p(\boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{\beta}, \boldsymbol{w}) = \prod_k \mathcal{G}am(\alpha_k \mid a, b)\mathcal{D}ir(\boldsymbol{\beta}_k \mid \eta) \qquad (1)$$
$$\prod_d \mathcal{D}ir(\theta_d \mid \boldsymbol{\alpha}) \prod_n \mathcal{M}ult(z_{dn} \mid \theta_d)\mathcal{M}ult(w_{dn} \mid \boldsymbol{\beta}_{z_{dn}})$$

Many inference methods have been developed for this model; the approach upon which we

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 293–296,
Suntec, Singapore, 4 August 2009.

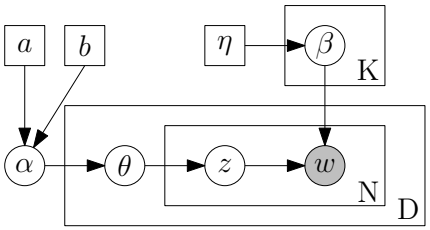


Figure 1: Graphical model for LDA.

build is the collapsed Gibbs sampler (Griffiths and Steyvers, 2006). Here, the random variables $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ are analytically integrated out. The main sampling variables are the z_{dn} indicators (as well as the hyperparameters: η and a, b). The conditional distribution for z_{dn} conditioned on all other variables in the model gives the following Gibbs sampling distribution $p(z_{dn} = k)$:

$$\frac{\#_{z=k}^{-dn} + \alpha_k}{\sum_{k'}(\#_{z=k'}^{-dn} + \alpha_{k'})} \frac{\#_{z=k,w=w_{dn}}^{-dn} + \eta}{\sum_{k'}(\#_{z=k',w=w_{dn}}^{-dn} + \eta)} \quad (2)$$

Here, $\#_{\chi}^{-dn}$ denotes the number of times event χ occurs in the entire corpus, excluding word n in document d. Intuitively, the first term is a (smoothed) relative frequency of topic k occurring; the second term is a (smoothed) relative frequency of topic k giving rise to word w_{dn}.

A Markov random field specifies a joint distribution over a collection of random variables $x_1, \ldots, x_N$. An undirected graph structure stipulates how the joint distribution factorizes over these variables. Given a graph $\mathcal{G} = (V, E)$, where $V = \{x_1, \ldots, x_N\}$, let $\mathcal{C}$ denote a subset of all the cliques of $\mathcal{G}$. Then, the MRF specifies the joint distribution as: $p(\boldsymbol{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(\boldsymbol{x}_c)$. Here, $Z = \sum_{\boldsymbol{x}} \prod_{c \in \mathcal{C}} \psi_c(\boldsymbol{x}_c)$ is the partition function, $\boldsymbol{x}_c$ is the subset of $\boldsymbol{x}$ contained in clique c and ψ_c is *any* non-negative function that measures how "good" a particular configuration of variables $\boldsymbol{x}_c$ is. The ψs are called potential functions.

3 Markov Random Topic Fields

Suppose that we have access to a collection of documents, but do not believe that these documents are all independent. In this case, the generative story of LDA no longer makes sense: related documents are more likely to have "similar" topic structures. For instance, in the scientific community, if paper A cites paper B, we would (a priori) expect the topic distributions for papers A and B to be related. Similarly, if two papers share an author, we might expect them to be topically related.

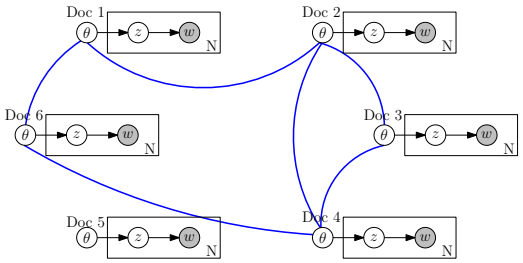


Figure 2: Example Markov Random Topic Field (variables α and $\boldsymbol{\beta}$ are excluded for clarify).

Of if they are both published at EMNLP. Or if they are published in the same year, or come out of the same institution, or many other possibilities.

Regardless of the source of this notion of similarity, we suppose that we can represent the relationship between documents in the form of a graph $\mathcal{G} = (V, E)$. The vertices in this graph are the documents and the edges indicate relatedness. Note that the resulting model will not be fully generative, but is still probabilistically well defined.

3.1 Single Graph

There are multiple possibilities for augmenting LDA with such graph structure. We could "link" the topic distributions $\boldsymbol{\theta}$ over related documents; we could "like" the topic indicators $\boldsymbol{z}$ over related documents. We consider the former because it leads to a more natural model. The idea is to "unroll" the D-plate in the graphical model for LDA (Figure 1) and connect (via undirected links) the $\boldsymbol{\theta}$ variables associated with connected documents. Figure 2 shows an example MRTF over six documents, with thick edges connecting the $\boldsymbol{\theta}$ variables of "related" documents. Note that each $\boldsymbol{\theta}$ still has α as a parent and each w has $\boldsymbol{\beta}$ as a parent: these are left off for figure clarity.

The model is a straightforward "integration" of LDA and an MRF specified by the document relationships $\mathcal{G}$. We begin with the joint distribution specified by LDA (see Eq (1)) and add in edge potentials for each edge in the document graph $\mathcal{G}$ that "encourage" the topic distributions of neighboring documents to be similar. The potentials all have the form:

$$\psi_{d,d'}(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'}) = \exp\left[-\ell_{d,d'} \rho(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'})\right] \quad (3)$$

Here, $\ell_{d,d'}$ is a "measure of strength" of the importance of the connection between d and d' (and will be inferred as part of the model). ρ is a distance metric measuring the dissimilarity between $\boldsymbol{\theta}_d$ and $\boldsymbol{\theta}_{d'}$. For now, this is Euclidean distance

(i.e., $\rho(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'}) = ||\boldsymbol{\theta}_d - \boldsymbol{\theta}_{d'}||$); later, we show that alternative distance metrics are preferable.

Adding the graph structure necessitates the addition of hyperparameters ℓ_e for every edge $e \in E$. We place an exponential prior on each $1/\ell_e$ with parameter λ: $p(\ell_e \mid \lambda) = \lambda \exp(-\lambda/\ell_e)$. Finally, we place a vague $\mathcal{G}am(\lambda_a, \lambda_b)$ prior on λ.

3.2 Multiple Graphs

In many applications, there may be multiple graphs that apply to the same data set, $\mathcal{G}_1, \ldots, \mathcal{G}_J$. In this case, we construct a single MRF based on the union of these graph structures. Each edge now has L-many parameters (one for each graph j) ℓ^j_e. Each graph also has its own exponential prior parameter λ_j. Together, this yields:

$$\psi_{d,d'}(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'}) = \exp\left[-\sum_j \ell^j_{d,d'} \rho(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'})\right] \quad (4)$$

Here, the sum ranges only over those graphs that have (d, d') in their edge set.

4 Inference

Inference in MRTFs is somewhat complicated from inference in LDA, due to the introduction of the additional potential functions. In particular, while it is possible to analytically integrate out $\boldsymbol{\theta}$ in LDA (due to multinomial/Dirichlet conjugacy), this is no longer possible in MRTFs. This means that we must explicitly represent (and sample over) the topic distributions $\boldsymbol{\theta}$ in the MRTF.

This means that we must sample over the following set of variables: $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$, $\boldsymbol{z}$, $\boldsymbol{\ell}$ and λ. Sampling for α remains unchanged from the LDA case. Sampling for variables *except* $\boldsymbol{\theta}$ is easy:

$$z_{dn} = k: \quad \theta_{dk} \frac{\#^{-dn}_{z=k, w=w_{dn}} + \eta}{\sum_{k'} (\#^{-dn}_{z=k', w=w_{dn}} + \eta)} \quad (5)$$

$$1/\ell_{d,d'} \sim \mathcal{E}xp\Big(\lambda + \rho(\boldsymbol{\theta}_d, \boldsymbol{\theta}_{d'})\Big) \quad (6)$$

$$\lambda \sim \mathcal{G}am\Big(\lambda_a + |E|, \lambda_b + \sum_e \ell_e\Big) \quad (7)$$

The latter two follow from simple conjugacy. When we use multiple graphs, we assign a separate λ for each graph.

For sampling $\boldsymbol{\theta}$, we resort to a Metropolis-Hastings step. Our proposal distribution is the Dirichlet *posterior* over $\boldsymbol{\theta}$, given all the current assignments. The acceptance probability then just depends on the graph distances. In particular, once $\boldsymbol{\theta}_d$ is drawn from the posterior Dirichlet, the acceptance probability becomes $\prod_{d' \in \mathcal{N}(d)} \psi_{d,d'}$, where $\mathcal{N}(d)$ denotes the neighbors of d. For each document, we run 10 Metropolis steps; the acceptance rates are roughly 25%.

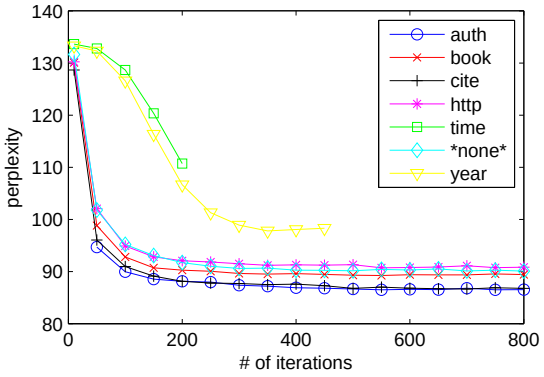


Figure 3: Held-out perplexity for different graphs.

5 Experiments

Our experiments are on a collection for 7441 document abstracts crawled from CiteSeer. The crawl was seeded with a collection of ten documents from each of: ACL, EMNLP, SIGIR, ICML, NIPS, UAI. This yields 650 thousand words of text after remove stop words. We use the following graphs (number in parens is the number of edges):

auth: shared author ($47k$)
book: shared booktitle/journal ($227k$)
cite: one cites the other ($18k$)
http: source file from same domain ($147k$)
time: published within one year ($4122k$)
year: published in the same year ($2101k$)

Other graph structures are of course possible, but these were the most straightforward to cull.

The first thing we look at is convergence of the samplers for the different graphs. See Figure 3. Here, we can see that the author graph and the citation graph provide improved perplexity to the straightforward LDA model (called "*none*"), and that convergence occurs in a few hundred iterations. Due to their size, the final two graphs led to significantly slower inference than the first four, so results with those graphs are incomplete.

Tuning Graph Parameters. The next item we investigate is whether it is important to tune the graph connectivity weights (the ℓ and λ variables). It turns out this is *incredibly* important; see Figure 4. This is the same set of results as Figure 3, but without ℓ and λ tuning. We see that the graph-based methods *do not* improve over the baseline.

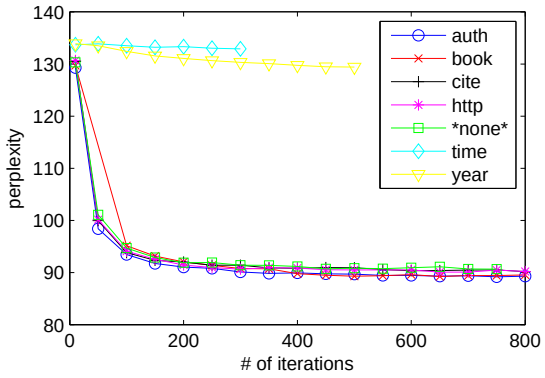


Figure 4: Held-out perplexity for difference graph structures without graph parameter tuning.

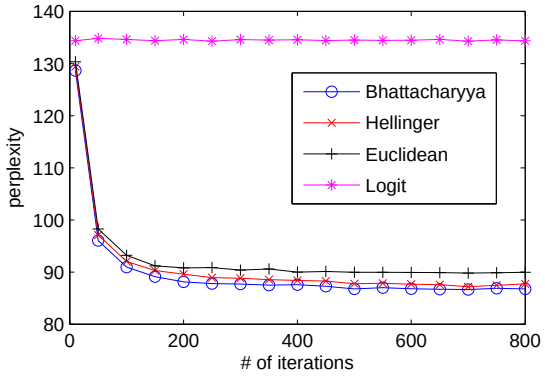


Figure 5: Held-out perplexity for different distance metrics.

Distance Metric. Next, we investigate the use of different distance metrics. We experiments with Bhattacharyya, Hellinger, Euclidean and logistic-Euclidean. See Figure 5 (this is just for the auth graph). Here, we see that Bhattacharyya and Hellinger (well motivated distances for probability distributions) outperform the Euclidean metrics.

Using Multiple Graphs Finally, we compare results using combinations of graphs. Here, we run every sampler for 500 iterations and compute standard deviations based on ten runs (year and time are excluded). The results are in Table 1. Here, we can see that adding graphs (almost) always helps and never hurts. By adding all the graphs together, we are able to achieve an absolute reduction in perplexity of 9 points (roughly 10%). As discussed, this hinges on the tuning of the graph parameters to allow different graphs to have different amounts of influence.

6 Discussion

We have presented a graph-augmented model for topic models and shown that a simple combined Gibbs/MH sampler is efficient in these models.

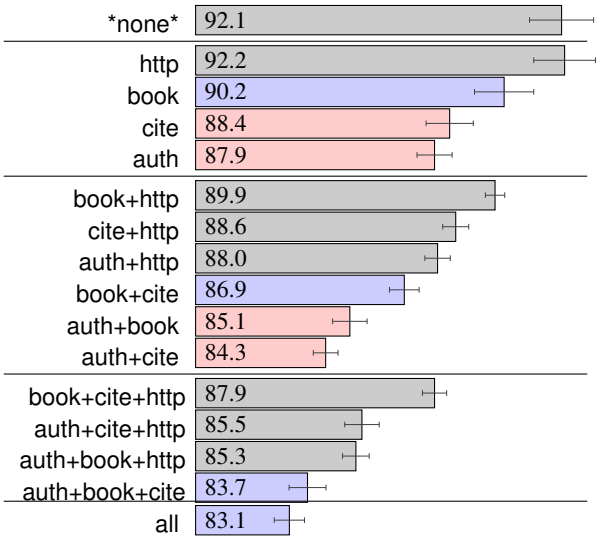


Table 1: Comparison of held-out perplexities for varying graph structures with two standard deviation error bars; grouped by number of graphs. Grey bars are indistinguishable from best model in previous group; blue bars are at least two stddevs better; red bars are at least four stddevs better.

Using data from the scientific domain, we have shown that we can achieve significant reductions in perplexity on held-out data using these models. Our model resembles recent work on hypertext topic models (Gruber et al., 2008; Sun et al., 2008) and blog influence (Nallapati and Cohen, 2008), but is specifically tailored toward undirected models. Ours is an alternative to the recently proposed Markov Topic Models approach (Wang et al., 2009). While the goal of these two models is similar, the approaches differ fairly dramatically: we use the graph structure to inform the per-document topic distributions; they use the graph structure to inform the unigram models associated with each topic. It would be worthwhile to directly compare these two approaches.

References

David Blei, Andrew Ng, and Michael Jordan. 2003. Latent Dirichlet allocation. *JMLR*, 3.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6).

Tom Griffiths and Mark Steyvers. 2006. Probabilistic topic models. In *Latent Semantic Analysis: A Road to Meaning*.

Amit Gruber, Michal Rosen-Zvi, , and Yair Weiss. 2008. Latent topic models for hypertext. In *UAI*.

Ramesh Nallapati and William Cohen. 2008. Link-PLSA-LDA: A new unsupervised model for topics and influence of blogs. In *Conference for Webblogs and Social Media*.

Congkai Sun, Bin Gao, Zhenfu Cao, and Hang Li. 2008. HTM: A topic model for hypertexts. In *EMNLP*.

Chong Wang, Bo Thiesson, Christopher Meek, and David Blei. 2009. Markov topic models. In *AI-Stats*.

Multi-Document Summarization using Sentence-based Topic Models

Dingding Wang [1] **Shenghuo Zhu** [2] **Tao Li** [1] **Yihong Gong** [2]
1. School of Computer Science, Florida International University, Miami, FL, 33199
2. NEC Laboratories America, Cupertino, CA 95014, USA.
{dwang003,taoli}@cs.fiu.edu {zsh,ygong}@sv.nec-labs.com

Abstract

Most of the existing multi-document summarization methods decompose the documents into sentences and work directly in the sentence space using a term-sentence matrix. However, the knowledge on the document side, i.e. the topics embedded in the documents, can help the context understanding and guide the sentence selection in the summarization procedure. In this paper, we propose a new Bayesian sentence-based topic model for summarization by making use of both the term-document and term-sentence associations. An efficient variational Bayesian algorithm is derived for model parameter estimation. Experimental results on benchmark data sets show the effectiveness of the proposed model for the multi-document summarization task.

1 Introduction

With the continuing growth of online text resources, document summarization has found wide-ranging applications in information retrieval and web search. Many multi-document summarization methods have been developed to extract the most important sentences from the documents. These methods usually represent the documents as term-sentence matrices (where each row represents a sentence and each column represents a term) or graphs (where each node is a sentence and each edge represents the pairwise relationship among corresponding sentences), and ranks the sentences according to their scores calculated by a set of predefined features, such as term frequency-inverse sentence frequency (TF-ISF) (Radev et al., 2004; Lin and Hovy, 2002), sentence or term position (Yih et al., 2007), and number of keywords (Yih et al., 2007). Typical existing summarization methods include centroid-based methods (e.g., MEAD (Radev et al., 2004)), graph-ranking based methods (e.g., LexPageRank (Erkan and Radev, 2004)), non-negative matrix factorization (NMF) based methods (e.g., (Lee and Seung, 2001)), Conditional random field (CRF) based summarization (Shen et al., 2007), and LSA based methods (Gong and Liu, 2001).

There are two limitations with most of the existing multi-document summarization methods: (1) They work directly in the sentence space and many methods treat the sentences as independent of each other. Although few work tries to analyze the context or sequence information of the sentences, the document side knowledge, i.e. the topics embedded in the documents are ignored. (2) Another limitation is that the sentence scores calculated from existing methods usually do not have very clear and rigorous probabilistic interpretations. Many if not all of the sentence scores are computed using various heuristics as few research efforts have been reported on using generative models for document summarization.

In this paper, to address the above issues, we propose a new Bayesian sentence-based topic model for multi-document summarization by making use of both the term-document and term-sentence associations. Our proposal explicitly models the probability distributions of selecting sentences given topics and provides a principled way for the summarization task. An efficient variational Bayesian algorithm is derived for estimating model parameters.

2 Bayesian Sentence-based Topic Models (BSTM)

2.1 Model Formulation

The entire document set is denoted by $\mathcal{D}$. For each document $d \in \mathcal{D}$, we consider its unigram language model,

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 297–300,
Suntec, Singapore, 4 August 2009.

$$p(W_1^n|\boldsymbol{\theta}_d) = \prod_{i=1}^{n} p(W_i|\boldsymbol{\theta}_d),$$

where $\boldsymbol{\theta}_d$ denotes the model parameter for document d, W_1^n denotes the sequence of words $\{W_i \in \mathcal{W}\}_{i=1}^n$, i.e. the content of the document. $\mathcal{W}$ is the vocabulary. As topic models, we further assume the unigram model as a mixture of several topic unigram models,

$$p(W_i|\boldsymbol{\theta}_d) = \sum_{T_i \in \mathcal{T}} p(W_i|T_i)p(T_i|\boldsymbol{\theta}_d),$$

where $\mathcal{T}$ is the set of topics. Here, we assume that given a topic, generating words is independent from the document, i.e.

$$p(W_i|T_i, \boldsymbol{\theta}_d) = p(W_i|T_i).$$

Instead of freely choosing topic unigram models, we further assume that topic unigram models are mixtures of some existing *base unigram models*, i.e.

$$p(W_i|T_i) = \sum_{s \in \mathcal{S}} p(W_i|S_i = s)p(S_i = s|T_i),$$

where $\mathcal{S}$ is the set of base unigram models. Here, we use sentence language models as the base models. One benefit of this assumption is that each topic is represented by meaningful sentences, instead of directly by keywords. Thus we have

$$p(W_i|\boldsymbol{\theta}_d) = \sum_{t \in \mathcal{T}} \sum_{s \in \mathcal{S}} p(W_i|S_i = s)p(S_i = s|T_i = t)p(T_i = t|\boldsymbol{\theta}_d).$$

Here we use parameter U_{st} for the probability of choosing base model s given topic t, $p(S_i = s|T_i = t) = U_{st}$, where $\sum_s U_{st} = 1$. We use parameters $\{\boldsymbol{\theta}_d\}$ for the probability of choosing topic t given document d, where $\sum_t \Theta_{dt} = 1$. We assume that the parameters of base models, $\{B_{ws}\}$, are given, i.e. $p(W_i = w|S_i = s) = B_{ws}$, where $\sum_w B_{ws} = 1$. Usually, we obtain B_{ws} by empirical distribution words of sentence s.

2.2 Parameter Estimation

For summarization task, we concern how to describe each topic with the given sentences. This can be answered by the parameter of choosing base model s given topic t, U_{st}. Comparing to parameter U_{st}, we concern less about the topic distribution of each document, i.e. Θ_{dt}. Thus we choose Bayesian framework to estimate U_{st} by marginalizing Θ_{dt}. To do so, we assume a Dirichlet prior for $\boldsymbol{\Theta}_{d\cdot} \sim \mathrm{Dir}(\boldsymbol{\alpha})$, where vector $\boldsymbol{\alpha}$ is a hyperparameter. Thus the likelihood is

$$\begin{aligned} f(\mathbf{U};\mathbf{Y}) &= \prod_d \int \prod_i p(Y_{id}|\boldsymbol{\theta}_d)\pi(\boldsymbol{\theta}_d|\boldsymbol{\alpha})d\boldsymbol{\theta}_d \\ &= B(\boldsymbol{\alpha})^{-D} \int \prod_{id} [\mathbf{B}\mathbf{U}\boldsymbol{\Theta}^\top]_{id}^{Y_{id}} \times \prod_{dk} \Theta_{dk}^{\alpha_k - 1} d\,\boldsymbol{\Theta}. \end{aligned} \tag{1}$$

As Eq. (1) is intractable, LDA (Blei et al., 2001) applies variational Bayesian, which is to maximize a variational bound of the integrated likelihood. Here we write the variational bound.

Definition 1 *The variational bound is*

$$\widetilde{f}(\mathbf{U},\mathbf{V};\mathbf{Y}) = \prod_d \frac{B(\boldsymbol{\alpha}+\boldsymbol{\gamma}_{d,\cdot})}{B(\boldsymbol{\alpha})} \prod_{vkwd} \left(\frac{B_{wv}U_{vk}}{\phi_{vk;wd}}\right)^{Y_{wd}\phi_{vk;wd}} \tag{2}$$

where the domain of $\mathbf{V}$ *is*$\mathcal{V} = \{\mathbf{V} \in \mathbb{R}_+^{D\times K} : \sum_k V_{dk} = 1\}$, $\phi_{vk;wd} = B_{wv}U_{vk}V_{dk}/[\mathbf{B}\mathbf{U}\mathbf{V}^\top]_{wd}$, $\gamma_{dk} = \sum_{wv} Y_{wd}\phi_{vk;wd}$.

We have the following proposition.

Proposition 1 $f(\mathbf{U};\mathbf{Y}) \geq \sup_{\mathbf{V}\in\mathcal{V}} \widetilde{f}(\mathbf{U},\mathbf{V};\mathbf{Y})$.

Actually the optimum of this variational bound is the same as that obtained variational Bayesian approach. Due to the space limit, the proof of the proposition is omitted.

3 The Iterative Algorithm

The LDA algorithm (Blei et al., 2001) employed the variational Bayesian paradigm, which estimates the optimal variation bound for each $\mathbf{U}$. The algorithm requires an internal Expectation-Maximization (EM) procedure to find the optimal variational bound. The nested EM slows down the optimization procedure. To avoid the internal EM loop, we can directly optimize the variational bound to obtain the update rules.

3.1 Algorithm Derivation

First, we define the concept of Dirichlet adjustment, which is used in the algorithm for variational update rules involving Dirichlet distribution. Then, we define some notations for the update rules.

Definition 2 *We call vector* $\mathbf{y}$ *of size* K *is the* Dirichlet adjustment *of vector* $\mathbf{x}$ *of size* K *with respect to Dirichlet distribution* $D_K(\boldsymbol{\alpha})$ *if*

$$y_k = \exp(\Psi(\alpha_k + x_k) - \Psi(\sum_l (\alpha_l + x_l))),$$

where $\Psi(\cdot)$ *is digamma function. We denote it by* $\mathbf{y} = \mathcal{P}_D(\mathbf{x};\boldsymbol{\alpha})$.

We denote element-wise product of matrix $\mathbf{X}$ and matrix $\mathbf{Y}$ by $\mathbf{X} \circ \mathbf{Y}$, element-wise division by $\frac{\mathbf{X}}{\mathbf{Y}}$, obtaining $\mathbf{Y}$ via normalizing of each column of $\mathbf{X}$ as $\mathbf{Y} \overset{1}{\leftarrow} \mathbf{X}$, and obtaining $\mathbf{Y}$ via Dirichlet adjustment $\mathcal{P}_D(\cdot;\boldsymbol{\alpha})$ and normalization of each row of $\mathbf{X}$ as $\mathbf{Y} \overset{\mathcal{P}_D(\cdot;\boldsymbol{\alpha}),2}{\leftarrow} \mathbf{X}$, i.e., $\mathbf{z} = \mathcal{P}_D((\mathbf{X}_{d,\cdot})^\top;\boldsymbol{\alpha})$ and $Y_{d,k} = z_k/\sum_k z_k$. The following is the update rules for LDA:

$$\mathbf{U} \overset{1}{\leftarrow} \mathbf{B}^\top \left[\frac{\mathbf{Y}}{\mathbf{B}\widetilde{\mathbf{U}}\widetilde{\mathbf{V}}^\top}\right] \widetilde{\mathbf{V}} \circ \widetilde{\mathbf{U}} \tag{3}$$

$$\mathbf{V} \overset{\mathcal{P}_D(\cdot;\boldsymbol{\alpha}),2}{\leftarrow} \left[\frac{\mathbf{Y}}{\mathbf{B}\widetilde{\mathbf{U}}\widetilde{\mathbf{V}}^\top}\right]^\top (\mathbf{B}\mathbf{U}) \circ \widetilde{\mathbf{V}} \tag{4}$$

Algorithm 1 Iterative Algorithm

Input: $\mathbf{Y}$: term-document matrix
$\mathbf{B}$: term-sentence matrix
K : the number of latent topics
Output: $\mathbf{U}$: sentence-topic matrix
$\mathbf{V}$: auxiliary document-topic matrix

1: Randomly initialize $\mathbf{U}$ and $\mathbf{V}$, and normalize them
2: **repeat**
3: Update $\mathbf{U}$ using Eq. (3);
4: Update $\mathbf{V}$ using Eq. (4);
5: Compute $\widetilde{f}$ using Eq. (2);
6: **until** $\widetilde{f}$ converges.

3.2 Algorithm Procedure

The detail procedure is listed as Algorithm 1. ¿From the sentence-topic matrix $\mathbf{U}$, we include the sentence with the highest probability in each topic into the summary.

4 Relations with Other Models

In this section, we discuss the connections and differences of our BSTM model with two related models.

Recently, a new language model, factorization with sentence bases (FGB) (Wang et al., 2008) is proposed for document clustering and summarization by making use of both term-document matrix $\mathbf{Y}$ and term-sentence matrix $\mathbf{B}$. The FGB model computes two matrices $\mathbf{U}$ and $\mathbf{V}$ by optimizing

$$\mathbf{U}, \mathbf{V} = \arg\min_{\mathbf{U},\mathbf{V}} \ell(\mathbf{U}, \mathbf{V}),$$

where

$$\ell(\mathbf{U}, \mathbf{V}) = \mathsf{KL}\left(\mathbf{Y} \| \mathbf{B}\mathbf{U}\mathbf{V}^\top\right) - \ln \Pr(\mathbf{U}, \mathbf{V}).$$

Here, Kullback-Leibler divergence is used to measure the difference between the distributions of $\mathbf{Y}$ and the estimated $\mathbf{B}\mathbf{U}\mathbf{V}^\top$. Our BSTM is similar to the FGB summarization since they are all based on sentence-based topic model. The difference is that the document-topic allocation $\mathbf{V}$ is marginalized out in BSTM. The marginalization increases the stability of the estimation of the sentence-topic parameters. Actually, from the algorithm we can see that the difference lies in the Dirichlet adjustment. Experimental results show that our BSTM achieves better summarization results than FGB model.

Our BSTM model is also related to 3-factor non-negative matrix factorization (NMF) model (Ding et al., 2006) where the problem is to solve $\mathbf{U}$ and $\mathbf{V}$ by minimizing

$$\ell_F(\mathbf{U}, \mathbf{V}) = \|\mathbf{Y} - \mathbf{B}\mathbf{U}\mathbf{V}^\top\|_F^2. \quad (5)$$

Both BSTM and NMF models are used for solving $\mathbf{U}$ and $\mathbf{V}$ and have similar multiplicative update rules. Note that if the matrix $\mathbf{B}$ is the identity matrix, Eq. (5) leads to the derivation of the NMF algorithm with Frobenius norm in (Lee and Seung, 2001). However, our BSTM model is a generative probabilistic model and makes use of Dirichlet adjustment. The results obtained in our model have clear and rigorous probabilistic interpretations that the NMF model lacks. In addition, by marginalizing out $\mathbf{V}$, our BSTM model leads to better summarization results.

5 Experimental Results

5.1 Data Set

To evaluate the summarization results empirically, we use the DUC2002 and DUC2004 data sets, both of which are open benchmark data sets from Document Understanding Conference (DUC) for generic automatic summarization evaluation. Table 1 gives a brief description of the data sets.

	DUC2002	DUC2004
number of document collections	59	50
number of documents in each collection	∼10	10
data source	TREC	TDT
summary length	200 words	665bytes

Table 1: Description of the data sets for multi-document summarization

Systems	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU
DUC Best	**0.49869**	**0.25229**	**0.46803**	**0.28406**
Random	0.38475	0.11692	0.37218	0.18057
Centroid	0.45379	0.19181	0.43237	0.23629
LexPageRank	0.47963	0.22949	0.44332	0.26198
LSA	0.43078	0.15022	0.40507	0.20226
NMF	0.44587	0.16280	0.41513	0.21687
KM	0.43156	0.15135	0.40376	0.20144
FGB	0.48507	0.24103	0.45080	0.26860
BSTM	**0.48812**	**0.24571**	**0.45516**	**0.27018**

Table 2: Overall performance comparison on DUC2002 data using ROUGE evaluation methods.

Systems	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU
DUC Best	**0.38224**	**0.09216**	**0.38687**	**0.13233**
Random	0.31865	0.06377	0.34521	0.11779
Centroid	0.36728	0.07379	0.36182	0.12511
LexPageRank	0.37842	0.08572	0.37531	0.13097
LSA	0.34145	0.06538	0.34973	0.11946
NMF	0.36747	0.07261	0.36749	0.12918
KM	0.34872	0.06937	0.35882	0.12115
FGB	0.38724	0.08115	0.38423	0.12957
BSTM	**0.39065**	**0.09010**	**0.38799**	**0.13218**

Table 3: Overall performance comparison on DUC2004 data using ROUGE evaluation methods.

5.2 Implemented Systems

We implement the following most widely used document summarization methods as the baseline systems to compare with our proposed BSTM method. (1) Random: The method selects sentences randomly for each document collection.

(2) Centroid: The method applies MEAD algorithm (Radev et al., 2004) to extract sentences according to the following three parameters: centroid value, positional value, and first-sentence overlap. (3) LexPageRank: The method first constructs a sentence connectivity graph based on cosine similarity and then selects important sentences based on the concept of eigenvector centrality (Erkan and Radev, 2004). (4) LSA: The method performs latent semantic analysis on terms by sentences matrix to select sentences having the greatest combined weights across all important topics (Gong and Liu, 2001). (5) NMF: The method performs non-negative matrix factorization (NMF) on terms by sentences matrix and then ranks the sentences by their weighted scores (Lee and Seung, 2001). (6) KM: The method performs K-means algorithm on terms by sentences matrix to cluster the sentences and then chooses the centroids for each sentence cluster. (7) FGB: The FGB method is proposed in (Wang et al., 2008).

5.3 Evaluation Measures

We use ROUGE toolkit (version 1.5.5) to measure the summarization performance, which is widely applied by DUC for performance evaluation. It measures the quality of a summary by counting the unit overlaps between the candidate summary and a set of reference summaries. The full explanation of the evaluation toolkit can be found in (Lin and E.Hovy, 2003). In general, the higher the ROUGE scores, the better summarization performance.

5.4 Result Analysis

Table 2 and Table 3 show the comparison results between BSTM and other implemented systems. From the results, we have the follow observations: (1) Random has the worst performance. The results of LSA, KM, and NMF are similar and they are slightly better than those of Random. Note that LSA and NMF provide continuous solutions to the same K-means clustering problem while LSA relaxes the non-negativity of the cluster indicator of K-means and NMF relaxes the orthogonality of the cluster indicator (Ding and He, 2004; Ding et al., 2005). Hence all these three summarization methods perform clustering-based summarization: they first generate sentence clusters and then select representative sentences from each sentence cluster. (2) The Centroid system outperforms clustering-based summarization methods in most cases. This is mainly because the Centroid based algorithm takes into account positional value and first-sentence overlap which are not used in clustering-based summarization. (3) LexPageRank outperforms Centroid. This is due to the fact that LexPageRank ranks the sentence using eigenvector centrality which implicitly accounts for information subsumption among all sentences (Erkan and Radev, 2004). (4) FGB performs better than LexPageRank. Note that FGB model makes use of both term-document and term-sentence matrices. Our BSTM model outperforms FGB since the document-topic allocation is marginalized out in BSTM and the marginalization increases the stability of the estimation of the sentence-topic parameters. (5) Our BSTM method outperforms all other implemented systems and its performance is close to the results of the best team in the DUC competition. Note that the good performance of the best team in DUC benefits from their preprocessing on the data using deep natural language analysis which is not applied in our implemented systems.

The experimental results provide strong evidence that our BSTM is a viable method for document summarization.

Acknowledgement: The work is partially supported by NSF grants IIS-0546280, DMS-0844513 and CCF-0830659.

References

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *Advances in Neural Information Processing Systems 14*.

C. Ding and X. He. K-means clustering and principal component analysis. In *Prodeedings of ICML 2004*.

Chris Ding, Xiaofeng He, and Horst Simon. 2005. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of Siam Data Mining*.

Chris Ding, Tao Li, Wei Peng, and Haesun Park. 2006. Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proceedings of SIGKDD 2006*.

G. Erkan and D. Radev. 2004. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of EMNLP 2004*.

Y. Gong and X. Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of SIGIR*.

Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*.

C-Y. Lin and E.Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of NLT-NAACL 2003*.

C-Y. Lin and E. Hovy. 2002. From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of ACL 2002*.

I. Mani. 2001. *Automatic summarization*. John Benjamins Publishing Company.

D. Radev, H. Jing, M. Stys, and D. Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing and Management*, pages 919–938.

B. Ricardo and R. Berthier. 1999. *Modern information retrieval*. ACM Press.

D. Shen, J-T. Sun, H. Li, Q. Yang, and Z. Chen. 2007. Document summarization using conditional random fields. In *Proceedings of IJCAI 2007*.

Dingding Wang, Shenghuo Zhu, Tao Li, Yun Chi, and Yihong Gong. 2008. Integrating clustering and multi-document summarization to improve document understanding. In *Proceedings of CIKM 2008*.

W-T. Yih, J. Goodman, L. Vanderwende, and H. Suzuki. 2007. Multi-document summarization by maximizing informative content-words. In *Proceedings of IJCAI 2007*.

Validating the web-based evaluation of NLG systems

Alexander Koller
Saarland U.
koller@mmci.uni-saarland.de

Kristina Striegnitz
Union College
striegnk@union.edu

Donna Byron
Northeastern U.
dbyron@ccs.neu.edu

Justine Cassell
Northwestern U.
justine@northwestern.edu

Robert Dale
Macquarie U.
Robert.Dale@mq.edu.au

Sara Dalzel-Job
U. of Edinburgh
S.Dalzel-Job@sms.ed.ac.uk

Jon Oberlander
U. of Edinburgh
{J.Oberlander|J.Moore}@ed.ac.uk

Johanna Moore
U. of Edinburgh

Abstract

The GIVE Challenge is a recent shared task in which NLG systems are evaluated over the Internet. In this paper, we validate this novel NLG evaluation methodology by comparing the Internet-based results with results we collected in a lab experiment. We find that the results delivered by both methods are consistent, but the Internet-based approach offers the statistical power necessary for more fine-grained evaluations and is cheaper to carry out.

1 Introduction

Recently, there has been an increased interest in evaluating and comparing natural language generation (NLG) systems on shared tasks (Belz, 2009; Dale and White, 2007; Gatt et al., 2008). However, this is a notoriously hard problem (Scott and Moore, 2007): Task-based evaluations with human experimental subjects are time-consuming and expensive, and corpus-based evaluations of NLG systems are problematic because a mismatch between human-generated output and system-generated output does not necessarily mean that the system's output is inferior (Belz and Gatt, 2008). This lack of evaluation methods which are both effective and efficient is a serious obstacle to progress in NLG research.

The GIVE Challenge (Byron et al., 2009) is a recent shared task which takes a third approach to NLG evaluation: By connecting NLG systems to experimental subjects over the Internet, it achieves a true task-based evaluation at a much lower cost. Indeed, the first GIVE Challenge acquired data from over 1100 experimental subjects online. However, it still remains to be shown that the results that can be obtained in this way are in fact comparable to more established task-based evaluation efforts, which are based on a carefully selected subject pool and carried out in a controlled laboratory environment. By accepting connections from arbitrary subjects over the Internet, the evaluator gives up control over the subjects' behavior, level of language proficiency, cooperativeness, etc.; there is also an issue of whether demographic factors such as gender might skew the results.

In this paper, we provide the missing link by repeating the GIVE evaluation in a laboratory environment and comparing the results. It turns out that where the two experiments both find a significant difference between two NLG systems with respect to a given evaluation measure, they always agree. However, the Internet-based experiment finds considerably more such differences, perhaps because of the higher number of experimental subjects ($n = 374$ vs. $n = 91$), and offers other opportunities for more fine-grained analysis as well. We take this as an empirical validation of the Internet-based evaluation of GIVE, and propose that it can be applied to NLG more generally. Our findings are in line with studies from psychology that indicate that the results of web-based experiments are typically consistent with the results of traditional experiments (Gosling et al., 2004). Nevertheless, we do find and discuss some effects of the uncontrolled subject pool that should be addressed in future Internet-based NLG challenges.

2 The GIVE Challenge

In the GIVE scenario (Byron et al., 2009), users try to solve a treasure hunt in a virtual 3D world that they have not seen before. The computer has complete information about the virtual world. The challenge for the NLG system is to generate, in real time, natural-language instructions that will guide the users to the successful completion of their task.

From the perspective of the users, GIVE consists in playing a 3D game which they start from a website. The game displays a virtual world and allows the user to move around in the world and manipulate objects; it also displays the generated

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 301–304,
Suntec, Singapore, 4 August 2009.

instructions. The first room in each game is a tutorial room in which users learn how to interact with the system; they then enter one of three evaluation worlds, where instructions for solving the treasure hunt are generated by an NLG system. Players can either finish a game successfully, lose it by triggering an alarm, or cancel the game at any time.

When a user starts the game, they are randomly connected to one of the three worlds and one of the NLG systems. The GIVE-1 Challenge evaluated five NLG systems, which we abbreviate as A, M, T, U, and W below. A running GIVE NLG system has access to the current state of the world and to an automatically computed plan that tells it what actions the user should perform to solve the task. It is notified whenever the user performs some action, and can generate an instruction and send it to the client for display at any time.

3 The experiments

The web experiment. For the GIVE-1 challenge, 1143 valid games were collected over the Internet over the course of three months. These were distributed over three evaluation worlds (World 1: 374, World 2: 369, World 3: 400). A game was considered valid if the game client didn't crash, the game wasn't marked as a test run by the developers, and the player completed the tutorial.

Of these games, 80% were played by males and 10% by females (the remaining 10% of the participants did not specify their gender). The players were widely distributed over countries: 37% connected from IP addresses in the US, 33% from Germany, and 17% from China; the rest connected from 45 further countries. About 34% of the participants self-reported as native English speakers, and 62% specified a language proficiency level of at least "expert" (3 on a 5-point scale).

The lab experiment. We repeated the GIVE-1 evaluation in a traditional laboratory setting with 91 participants recruited from a college campus. In the lab, each participant played the GIVE game once with each of the five NLG systems. To avoid learning effects, we only used the first game run from each subject in the comparison with the web experiment; as a consequence, subjects were distributed evenly over the NLG systems. To accommodate for the much lower number of participants, the laboratory experiment only used a single game world – World 1, which was known from the online version to be the easiest world.

Among this group of subjects, 93% self-rated their English proficiency as "expert" or better; 81% were native speakers. In contrast to the online experiment, 31% of participants were male and 65% were female (4% did not specify their gender).

Results: Objective measures. The GIVE software automatically recorded data for five objective measures: the percentage of successfully completed games and, for the successfully completed games, the number of instructions generated by the NLG system, of actions performed by the user (such as pushing buttons), of steps taken by the user (i.e., actions plus movements), and the task completion time (in seconds).

Fig. 1 shows the results for the objective measures collected in both experiments. To make the results comparable, the table for the Internet experiment only includes data for World 1. The task success rate is only evaluated on games that were completed successfully or lost, not cancelled, as laboratory subjects were asked not to cancel. This brings the number of Internet subjects to 322 for the success rate, and to 227 (only successful games) for the other measures.

Task success is the percentage of successfully completed games; the other measures are reported as means. The chart assigns systems to groups A through C or D for each evaluation measure. Systems in group A are better than systems in group B, and so on; if two systems have no letter in common, the difference between them is significant with $p < 0.05$. Significance was tested using a χ^2-test for task success and ANOVAs for instructions, steps, actions, and seconds. These were followed by post hoc tests (pairwise χ^2 and Tukey) to compare the NLG systems pairwise.

Results: Subjective measures. Users were asked to fill in a questionnaire collecting subjective ratings of various aspects of the instructions. For example, users were asked to rate the overall quality of the direction giving system (on a 7-point scale), the choice of words and the referring expressions (on 5-point scales), and they were asked whether they thought the instructions came at the right time. Overall, there were twelve subjective measures (see (Byron et al., 2009)), of which we only present four typical ones for space reasons.

For each question, the user could choose not to answer. On the Internet, subjects made considerable use of this option: for instance, 32% of users

	Objective Measures										Subjective Measures							
	task success		instructions		steps		actions		seconds		overall		choice of words		referring expressions		timing	
A	91%	A	83.4	B	99.8	A	9.4	A	123.9	A	4.7	A	4.7	A	4.7	A	81%	A
M	76%	B	68.1	A	145.1	B	10.0	AB	195.4	BC	3.8	AB	3.8	B	4.0	B	70%	ABC
T	85%	AB	97.8	C	142.1	B	9.7	AB	174.4	B	4.4	B	4.4	AB	4.3	AB	73%	AB
U	93%	AB	99.8	C	142.6	B	10.3	B	194.0	BC	4.0	B	4.0	B	4.0	B	51%	C
W	24%	C	159.7	D	256.0	C	9.6	AB	234.1	C	3.8	AB	3.8	B	4.2	AB	50%	BC
A	100%	A	78.2	AB	93.4	A	9.9	A	143.9	A	5.7	A	4.7	A	4.8	A	92%	A B
M	95%	A	66.3	A	141.8	B	10.5	A	211.8	B	5.4	A	3.8	B	4.3	A	95%	A B
T	93%	A	107.2	CD	134.6	B	9.6	A	205.6	B	4.9	A	4.5	A B	4.4	A	64%	A B
U	100%	A	88.8	BC	128.8	B	9.8	A	195.1	AB	5.7	A	4.7	A	4.3	A	100%	A
W	17%	B	134.5	D	213.5	C	10.0	A	252.5	B	5.0	A	4.5	A B	4.0	A	100%	B

Figure 1: Objective and selected subjective measures on the web (top) and in the lab (bottom).

didn't fill in the "overall evaluation" field of the questionnaire. In the laboratory experiment, the subjects were asked to fill in the complete questionnaire and the response rate is close to 100%.

The results for the four selected subjective measures are summarized in Fig. 1 in the same way as the objective measures. Also as above, the table is based only on successfully completed games in World 1. We will justify this latter choice below.

	Web		
	games	reported	mean
success	227 = 61%	93%	4.9
lost	92 = 24%	48%	3.4
cancelled	55 = 15%	16%	3.3
	Lab		
	# games	reported	mean
success	73 = 80%	100%	5.4
lost	18 = 20%	94%	3.3
cancelled	0	–	–

Figure 2: Skewed results for "overall evaluation".

4 Discussion

The primary question that interests us in a comparative evaluation is which NLG systems performed significantly better or worse on any given evaluation measure. In the experiments above, we find that of the 170 possible significant differences (= 17 measures $\times$ 10 pairs of NLG systems), the laboratory experiment only found six that the Internet-based experiment didn't find. Conversely, there are 26 significant differences that only the Internet-based experiment found. But even more importantly, all pairwise rankings are consistent across the two evaluations: Where both systems found a significant difference between two systems, they always ranked them in the same order. We conclude that the Internet experiment provides significance judgments that are comparable to, and in fact more precise than, the laboratory experiment.

Nevertheless, there are important differences between the laboratory and Internet-based results. For instance, the success rates in the laboratory tend to be higher, but so are the completion times. We believe that these differences can be attributed to the demographic characteristics of the participants. To substantiate this claim, we looked in some detail at differences in gender, language proficiency, and questionnaire response rates.

First, the gender distribution differed greatly between the Internet experiment (10% female) and the laboratory experiment (65% female). This is relevant because gender had a significant effect on task completion time (women took longer) and on six subjective measures including "overall evaluation" in the laboratory. We speculate that the difference in task completion time may be related to well-known gender differences in processing navigation instructions (Moffat et al., 1998).

Second, the two experiments collected data from subjects of different language proficiencies. While 93% of the participants in the laboratory experiment self-rated their English proficiency as "expert" or better, only 62% of the Internet participants did. This partially explains the lower task success rates on the Internet, as Internet subjects with English proficiencies of 3–5 performed significantly better on "task success" than the group with proficiencies 1–2. If we only look at the results of high-English-proficiency subjects on the Internet, the success rates for all NLG systems except W rise to at least 86%, and are thus close to the laboratory results.

Finally, the Internet data are skewed by the tendency of unsuccessful participants to not fill in the questionnaire. Fig. 2 summarizes some data about the "overall evaluation" question. Users who didn't complete the task successfully tended to judge the

systems much lower than successful users, but at the same time tended not to answer the question at all. This skew causes the mean subjective judgments across all Internet subjects to be artificially high. To avoid differences between the laboratory and the Internet experiment due to this skew, Fig. 1 includes only judgments from successful games.

In summary, we find that while the two experiments made consistent significance judgments, and the Internet-based evaluation methodology thus produces meaningful results, the absolute values they find for the individual evaluation measures differ due to the demographic characteristics of the participants in the two studies. This could be taken as a possible deficit of the Internet-based evaluation. However, we believe that the opposite is true. In many ways, an online user is in a much more natural communicative situation than a laboratory subject who is being discouraged from cancelling a frustrating task. In addition, every experiment – whether in the laboratory or on the Internet – suffers from some skew in the subject population due to sampling bias; for instance, one could argue that an evaluation that is based almost exclusively on native speakers in universities leads to overly benign judgments about the quality of NLG systems.

One advantage of the Internet-based approach to data collection over the laboratory-based one is that, due to the sheer number of subjects, we can detect such skews and deal with them appropriately. For instance, we might decide that we are only interested in the results from proficient English speakers and ignore the rest of the data; but we retain the option to run the analysis over all participants, and to analyze how much each system relies on the user's language proficiency. The amount of data also means that we can obtain much more fine-grained comparisons between NLG systems. For instance, the second and third evaluation world specifically exercised an NLG system's abilities to generate referring expressions and navigation instructions, respectively, and there were significant differences in the performance of some systems across different worlds. Such data, which is highly valuable for pinpointing specific weaknesses of a system, would have been prohibitively costly and time-consuming to collect with laboratory subjects.

5 Conclusion

In this paper, we have argued that carrying out task-based evaluations of NLG systems over the Internet is a valid alternative to more traditional laboratory-based evaluations. Specifically, we have shown that an Internet-based evaluation of systems in the GIVE Challenge finds consistent significant differences as a lab-based evaluation. While the Internet-based evaluation suffers from certain skews caused by the lack of control over the subject pool, it does find more differences than the lab-based evaluation because much more data is available. The increased amount of data also makes it possible to compare the quality of NLG systems across different evaluation worlds and users' language proficiency levels.

We believe that this type of evaluation effort can be applied to other NLG and dialogue tasks beyond GIVE. Nevertheless, our results also show that an Internet-based evaluation risks certain kinds of skew in the data. It is an interesting question for the future how this skew can be reduced.

References

A. Belz and A. Gatt. 2008. Intrinsic vs. extrinsic evaluation measures for referring expression generation. In *Proceedings of ACL-08:HLT, Short Papers*, pages 197–200, Columbus, Ohio.

A. Belz. 2009. That's nice ... what can you do with it? *Computational Linguistics*, 35(1):111–118.

D. Byron, A. Koller, K. Striegnitz, J. Cassell, R. Dale, J. Moore, and J. Oberlander. 2009. Report on the First NLG Challenge on Generating Instructions in Virtual Environments (GIVE). In *Proceedings of the 12th European Workshop on Natural Language Generation (Special session on Generation Challenges)*.

R. Dale and M. White, editors. 2007. *Proceedings of the NSF/SIGGEN Workshop for Shared Tasks and Comparative Evaluation in NLG*, Arlington, VA.

A. Gatt, A. Belz, and E. Kow. 2008. The TUNA challenge 2008: Overview and evaluation results. In *Proceedings of the 5th International Natural Language Generation Conference (INLG'08)*, pages 198–206.

S. D. Gosling, S. Vazire, S. Srivastava, and O. P. John. 2004. Should we trust Web-based studies? A comparative analysis of six preconceptions about Internet questionnaires. *American Psychologist*, 59:93–104.

S. Moffat, E. Hampson, and M. Hatzipantelis. 1998. Navigation in a "virtual" maze: Sex differences and correlation with psychometric measures of spatial ability in humans. *Evolution and Human Behavior*, 19(2):73–87.

D. Scott and J. Moore. 2007. An NLG evaluation competition? Eight reasons to be cautious. In *(Dale and White, 2007)*.

Extending a Surface Realizer to Generate Coherent Discourse

Eva Banik
The Open University
Milton Keynes, UK
e.banik@open.ac.uk

Abstract

We present a discourse-level Tree Adjoining Grammar which tightly integrates syntax and discourse levels, including a representation for discourse entities. We show that this technique makes it possible to extend an optimisation algorithm used in natural language generation (polarity filtering) to the discourse level. We implemented the grammar in a surface realizer and show that this technique can be used to reduce the search space by filtering out referentially incoherent solutions.

1 Introduction

A fundamental problem that microplanners and surface realizers face in natural language generation is how to restrict the search space of possible solutions. A traditional solution to this computational complexity problem is to divide the generation process into tractable sub-problems, each represented as a module in a pipeline, where every decision made by a module restricts the number of options available to others further down the line. Though such pipeline architectures are computationally efficient, they severely restrict the flexibility of the system and the quality of the generated output. Most systems with pipeline architectures generate relatively simple, domain-specific output. Systems that produce more complex linguistic constructions typically achieve this by adding more modules to the pipeline (e.g. a revision module (Robin, 1994) or aggregation (Shaw, 2002)). Since complex linguistic constructions often require interaction between modules, adding them to the repertoire of pipelined NLG systems becomes an engineering and programming task.

Integrated NLG systems have a simpler architecture because they do not need to model interactions between modules. However, they still face the problem of computational complexity that was originally solved by the pipeline model. Strategies that have been introduced to reduce the search space in integrated systems include greedy/incremental search algorithms (Stone et al., 2003), constructing a dependency graph for a flat semantic input and converting it into a derivation tree (Koller and Striegnitz, 2002), using planning algorithms (Appelt, 1985; Koller and Stone, 2007), polarity filtering (Kow, 2007) and using underspecified g-derivation trees (G-TAG, Danlos (2000)). Despite all these efforts, most systems still don't attempt to go above the sentence level or generate very complex sentences. In this paper we present a new technique for designing an integrated grammar for natural language generation. Using this technique it is possible to use linguistic constraints on referential coherence to automatically reduce the search space — which in turn makes it possible to generate longer and more coherent texts.

First we extend the grammar of a surface realizer to produce complex, multi-sentential output. Then we add a representation for discourse referents to the grammar, inspired by Centering Theory's notion of a backward looking center and preferred center. Having done this, we show that by integrating discourse-level representations into a syntactic grammar we can extend an optimization technique — polarity filtering (Kow, 2007; Gardent and Kow, 2006) — from syntactic realization to the discourse level.

2 The Problem of Referential Coherence

Referential coherence is the phenomenon which is responsible for the contrast in (1), in the sense that the example in (1b) is perceived to be more coherent than (1a).

(1) a Elixir is approved by the FDA. Viral skin disorders are relieved by

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 305–308,
Suntec, Singapore, 4 August 2009.

Aliprosan. Elixir is a white cream. Aliprosan is an ingredient of Elixir.

b Elixir is a white cream. Elixir is approved by the FDA. Elixir contains Aliprosan. Aliprosan relieves viral skin disorders.

Centering Theory (Grosz et al., 1995) is a frequently used framework for modeling referential coherence in discourse. It is based on the notion that for each utterance in a discourse there is a set of entities which are the *centers* of attention and which serve to link that utterance to other utterances in the same discourse segment. Entities mentioned by an utterance (the set of forward looking centers) form a partially ordered list called the Cf list where roughly, subjects are ranked highest, followed by objects, indirect objects and other arguments or adjuncts. The backward looking center of Un is said to be the most highly ranked element on the Cf list of Un-1 mentioned in the previous utterance.

Centering Theory has been adapted to NLG by Kibble (1999; 2001), and implemented in Kibble and Power (2004). Rather than using the notion of centering transitions as defined by Grosz et al. (1995), in these papers centering theory is redefined as constraints on **salience** and **cohesion**. These constraints state that there is a preference for consecutive utterances to keep the same center and that there is a preference for the center of Un to be realized as the highest ranked entity on the Cf list of Un. Kibble and Power (2004) show how these constraints can be used to drive text planning, sentence planning and pronominalization in an integrated fashion. Our approach is similar to Kibble and Power (2004) in that we don't use the concept of centering transitions. However, our method is more efficient in that Kibble and Power (2004) use centering transitions to rank the set of generated solutions (some of which are incoherent), whereas we encode centering constraints in elementary trees to reduce the search space of possible solutions *before* we start computing them.

3 GenI and Polarity Filtering

The grammar described in the next section was implemented in the GenI surface realizer (Kow, 2007), which uses a lexicalized feature-based Tree Adjoining Grammar to generate all possible paraphrases for a given flat semantic input. GenI implements an optimization technique called *polarity filtering* to constrain the effects of lexical ambiguity. The basic idea of polarity filtering is to associate elementary trees with a set of polarities. When these polarities don't 'cancel each other out', it means that it is not possible to combine the set of trees selected for a given input. This is a quick way to check whether the number of argument slots is the same as the number of potential arguments. For example, if the lexical selection consists of two trees for a given input, one of which provides an NP (`-NP`) and one of which expects two NPs (`-2NP`) then the sum of polarities will be `-NP` and therefore the generator will not attempt to combine the trees.

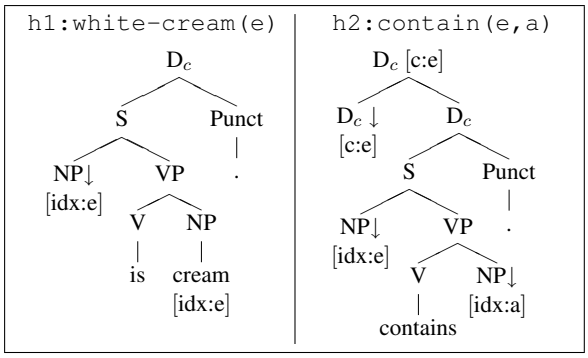


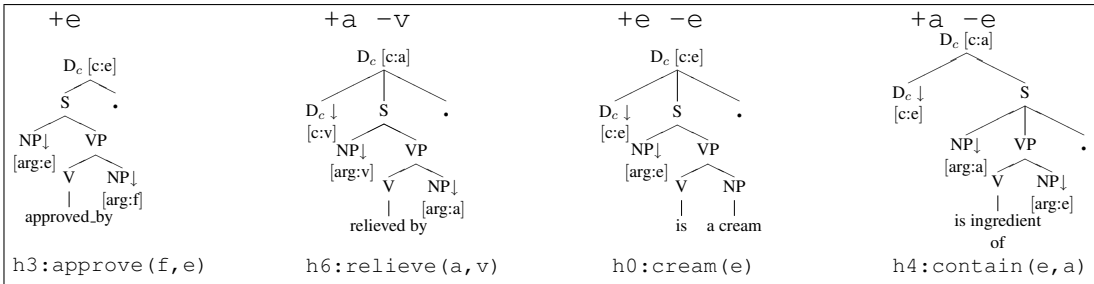
Figure 1: Elementary syntax/discourse trees

Values for polarities are defined as follows: every initial tree is assigned a `-cat` polarity for each substitution node of category *cat* and a `+cat` polarity if its root node is of category *cat*. Auxiliary trees are assigned a `-cat` polarity for each substitution node only.

Polarity filtering is a very powerful optimization technique, because it allows the generator to reduce the search space early on in the process, before it attempts to combine any trees.

4 An Integrated Syntax-Discourse Grammar

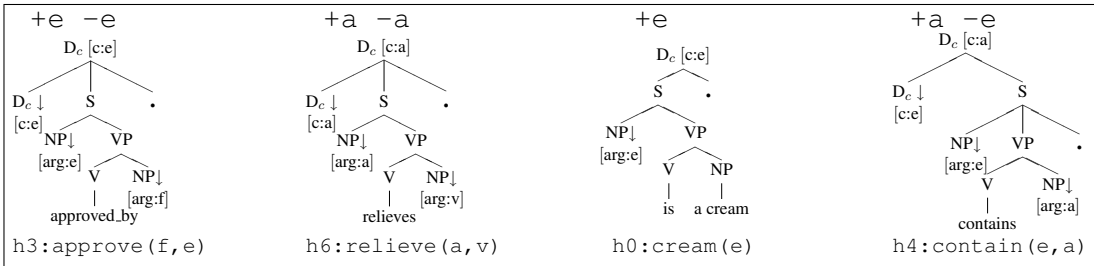
In order to generate mutisentential text, we first define a discourse-level Tree Adjoining Grammar. The trees in the grammar tightly integrate syntax and discourse representations in the sense that sentence-level elementary trees include one or more discourse-level nodes. The elementary trees in Fig. 1 illustrate what we mean by this: every lexical item that would normally project a sentence in a syntactic grammar (i.e., an S-rooted



+2a -v

Elixir is approved by the FDA. Viral skin disorders are relieved by Aliprosan. Elixir is a white cream. Aliprosan is an ingredient of Elixir.

Figure 2: Discourse-level polarities for (1a) sum up to +2a −v



+a

Elixir is a white cream. Elixir is approved by the FDA. Elixir contains Aliprosan. Aliprosan relieves viral skin disorders.

Figure 3: Discourse-level polarities for (1b) sum up to +a

tree) here projects a discourse clause (i.e., a Dc rooted tree). Every predicate that projects a discourse clause is assigned two kinds of elementary trees: a discourse initial tree (Fig. 1a) and a discourse continuing tree (Fig. 1b), which takes the preceding discourse clause as an argument.

We model referential coherence by associating a discourse entity with every root- and substitution node of category D_c. A discourse entity on a root node is "exported" by the elementary tree to be the center of attention in the next sentence. This roughly corresponds to Centering Theory's notion of a forward looking center. A discourse entity on a substitution node is the entity expected by the sentence to have been the center of attention in the previous utterance, roughly corresponding to the notion of backward looking center in Centering Theory.

For example, the tree on the left in Fig. 1. exports the discourse entity representing its subject ('e') as its "forward looking center". The tree on the right in Fig. 1. is looking for a discourse entity called 'e' as its "backward looking center" and exports the same discourse entity as its "forward looking center". The combination of these two trees therefore yields a coherent discourse, which is expected to be continued with an utterance centered on 'e'.

5 Polarity Filtering on Discourse Entities

By treating discourse entities on D_c nodes as an additional polarity key we can apply the polarity filtering technique on the discourse level. This means we can filter out lexical selections that wouldn't lead to a coherent discourse the same way as those lexical selections are filtered out which won't lead to a syntactically well formed sentence. To give an example, given the semantic representation in Figure 4 potential realizations by a generator which is not aware of discourse coherence would include both of the examples in (1).

As an experiment, we generated the above example using the same input but two different grammars. In the first case we used a grammar which consists of discourse-level trees but no annotations for discourse entities. The realizer pro-

```
h0:white_cream(e)
h1:elixir(e)
h2:fda(f)
h3:approve(f e)
h4:contain(e a)
h5:aliprosan(a)
h6:relieve(a v)
h7:viral_skin_disorders(v)
```

Figure 4: Input for the sentences in (1)

duced 192 solutions, including many incoherent ones such as (1a). In the second case, we used a grammar with the same trees, but annotated with discourse referents. In this case the realizer produced only 16 solutions, all of which maintained referential coherence. In the first case, the grammar provided 128 ways to associate trees with the input (tree sets), and the 192 solutions included all possible sentence orders. Since for most trees in the grammar there are more than one ways to annotate them with discourse referents, in the second case the grammar contained more trees (differing only in their discourse referent asignments). In this case there were 1536 tree sets selected for the same input. Of these, 1320 were discarded by polarity filtering on discourse entities. Of the remaining 216 tree sets 200 were ruled out by feature unification when the trees were combined.

Figures 2 and 3 illustrate two sets of trees that were selected by the realizer, corresponding to the examples in (1). Discourse-level polarity filtering in this example (for the input in (4)) discards all tree sets whose polarities don't sum up to one of the discourse entities, i.e., `+e`, `+a`, `+f` or `+v`. The polarity of the tree set in Fig.2 is `+2a -v` so the tree set is discarded. For the tree set in Fig.3 the polarities sum up to `+e` and the realizer attempts to combine the trees, which in this case leads to a referentially coherent solution (1b).

The search space of the realizer can be further restricted by only allowing tree sets whose polarities sum up to a specific discourse entity. In this case the realizer will produce paragraphs where the center of attention in the last sentence is the discourse entity used for polarity filtering.

6 Conclusions

We have described a discourse-level extension of Tree Adjoining Grammar which tightly integrates syntax with discourse and includes a representation of discourse entities. We have shown that including discourse entities in the grammar of a surface realizer improves the coherence of the generated text and that these variables can also be used in a very efficient optimization technique, polarity filtering, to filter out referentially incoherent solutions.

References

D.E. Appelt. 1985. *Planning English sentences*. Cambridge University Press, Cambridge.

L. Danlos. 2000. G-TAG: A lexicalized formalism for text generation inspired by Tree Adjoining Grammar. In A. Abeille and O. Rambow, editors, *Tree Adjoining Grammars: Formalisms, linguistic analysis and processing*, pages 343–370. CSLI, Stanford, CA.

C. Gardent and E. Kow. 2006. Three reasons to adopt TAG-based surface realisation. In *Proceedings of TAG+8)*, Sydney/Australia.

B.J. Grosz, A.K. Joshi, and S Weinstein. 1995. Centering: a framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

R. Kibble and R. Power. 2004. Optimizing referential coherence in text generation. *Computational Linguistics*, 30(4):401–416.

R. Kibble. 1999. Cb or not Cb? centering theory applied to NLG. In *ACL workshop on Discourse and Reference Structure*, pages 72–81.

R. Kibble. 2001. A reformulation of rule 2 of centering theory. *Comput. Linguist.*, 27(4):579–587.

A. Koller and M. Stone. 2007. Sentence generation as planning. In *Proceedings of ACL.*

A. Koller and K. Striegnitz. 2002. Generation as dependency parsing. In *Proceedings of ACL.*

E. Kow. 2007. *Surface realisation: ambiguity and determinism*. Ph.D. thesis, Universite de Henri Poincare - Nancy 1.

J. Robin. 1994. *Revision-based generation of Natural Language Summaries providing historical Background*. Ph.D. thesis, Columbia University.

J. Shaw. 2002. *Clause Aggregation: An approach to generating concise text*. Ph.D. thesis, Columbia University.

M. Stone, C. Doran, B. Webber, T. Bleam, and M. Palmer. 2003. Microplanning with communicative intentions: The SPUD system. *Computational Intelligence*, 19(4):311–381.

The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language

Rada Mihalcea
University of North Texas
rada@cs.unt.edu

Carlo Strapparava
FBK-IRST
strappa@fbk.eu

Abstract

In this paper, we present initial experiments in the recognition of deceptive language. We introduce three data sets of true and lying texts collected for this purpose, and we show that automatic classification is a viable technique to distinguish between truth and falsehood as expressed in language. We also introduce a method for class-based feature analysis, which sheds some light on the features that are characteristic for deceptive text.

You should not trust the devil, even if he tells the truth.
– Thomas of Aquin (medieval philosopher)

1 Introduction and Motivation

The discrimination between truth and falsehood has received significant attention from fields as diverse as philosophy, psychology and sociology. Recent advances in computational linguistics motivate us to approach the recognition of deceptive language from a data-driven perspective, and attempt to identify the salient features of lying texts using natural language processing techniques.

In this paper, we explore the applicability of computational approaches to the recognition of deceptive language. In particular, we investigate whether automatic classification techniques represent a viable approach to distinguish between truth and lies as expressed in written text. Although acoustic and other non-linguistic features were also found to be useful for this task (Hirschberg et al., 2005), we deliberately focus on written language, since it represents the type of data most frequently encountered on the Web (e.g., chats, forums) or in other collections of documents.

Specifically, we try to answer the following two questions. First, are truthful and lying texts separable, and does this property hold for different datasets? To answer this question, we use three different data sets that we construct for this purpose – consisting of true and false short statements on three different topics – and attempt to automatically separate them using standard natural language processing techniques.

Second, if truth and lies are separable, what are the distinctive features of deceptive texts? In answer to this second question, we attempt to identify some of the most salient features of lying texts, and analyse their occurrence in the three data sets.

The paper is organized as follows. We first briefly review the related work, followed by a description of the three data sets that we constructed. Next, we present our experiments and results using automatic classification, and introduce a method for the analysis of salient features in deceptive texts. Lastly, we conclude with a discussion and directions for future work.

2 Related Work

Very little work, if any, has been carried out on the automatic detection of deceptive language in written text. Most of the previous work has focused on the psychological or social aspects of lying, and there are only a few previous studies that have considered the linguistic aspects of falsehood.

In psychology, it is worthwhile mentioning the study reported in (DePaulo et al., 2003), where more than 100 cues to deception are mentioned. However, only a few of them are linguistic in nature, as e.g., word and phrase repetitions, while most of the cues involve speaker's behavior, including facial expressions, eye shifts, etc. (Newman et al., 2003) also report on a psycholinguistic study, where they conduct a qualitative analysis of true and false stories by using word counting tools.

Computational work includes the study of (Zhou et al., 2004), which studied linguistic cues for deception detection in the context of text-based asynchronous computer mediated communication, and (Hirschberg et al., 2005) who focused on deception in speech using primarily acoustic and prosodic features.

Our work is also related to the automatic classification of text genre, including work on author profiling (Koppel et al., 2002), humor recognition

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 309–312,
Suntec, Singapore, 4 August 2009.

TRUTH	LIE
ABORTION	
I believe abortion is not an option. Once a life has been conceived, it is precious. No one has the right to decide to end it. Life begins at conception,because without conception, there is no life.	A woman has free will and free choice over what goes on in her body. If the child has not been born, it is under her control. Often the circumstances an unwanted child is born into are worse than death. The mother has the responsibility to choose the best course for her child.
DEATH PENALTY	
I stand against death penalty. It is pompous of anyone to think that they have the right to take life. No court of law can eliminate all possibilities of doubt. Also, some circumstances may have pushed a person to commit a crime that would otherwise merit severe punishment.	Death penalty is very important as a deterrent against crime. We live in a society, not as individuals. This imposes some restrictions on our actions. If a person doesn't adhere to these restrictions, he or she forfeits her life. Why should taxpayers' money be spent on feeding murderers?
BEST FRIEND	
I have been best friends with Jessica for about seven years now. She has always been there to help me out. She was even in the delivery room with me when I had my daughter. She was also one of the Bridesmaids in my wedding. She lives six hours away, but if we need each other we'll make the drive without even thinking.	I have been friends with Pam for almost four years now. She's the sweetest person I know. Whenever we need help she's always there to lend a hand. She always has a kind word to say and has a warm heart. She is my inspiration.

Table 1: Sample true and deceptive statements

(Mihalcea and Strapparava, 2006), and others.

3 Data Sets

To study the distinction between true and deceptive statements, we required a corpus with explicit labeling of the truth value associated with each statement. Since we were not aware of any such data set, we had to create one ourselves. We focused on three different topics: opinions on abortion, opinions on death penalty, and feelings about the best friend. For each of these three topics an annotation task was defined using the Amazon Mechanical Turk service.

For the first two topics (*abortion* and *death penalty*), we provided instructions that asked the contributors to imagine they were taking part in a debate, and had 10-15 minutes available to express their opinion about the topic. First, they were asked to prepare a brief speech expressing their true opinion on the topic. Next, they were asked to prepare a second brief speech expressing the opposite of their opinion, thus lying about their true beliefs about the topic. In both cases, the guidelines asked for at least 4-5 sentences and as many details as possible.

For the third topic (*best friend*), the contributors were first asked to think about their best friend and describe the reasons for their friendship (including facts and anecdotes considered relevant for their relationship). Thus, in this case, they were asked to tell the truth about how they felt about their best friend. Next, they were asked to think about a person they could not stand, and describe it as if s/he were their best friend. In this second case, they had to lie about their feelings toward this person. As before, in both cases the instructions asked for at least 4-5 detailed sentences.

We collected 100 true and 100 false statements for each topic, with an average of 85 words per statement. Previous work has shown that data collected through the Mechanical Turk service is reliable and comparable in quality with trusted sources (Snow et al., 2008). We also made a manual verification of the quality of the contributions, and checked by hand the quality of all the contributions. With two exceptions – two entries where the true and false statements were identical, which were removed from the data – all the other entries were found to be of good quality, and closely following our instructions.

Table 1 shows an example of true and deceptive language for each of the three topics.

4 Experimental Setup and Results

For the experiments, we used two classifiers: Naïve Bayes and SVM, selected based on their performance and diversity of learning methodologies. Only minimal preprocessing was applied to the three data sets, which included tokenization and stemming. No feature selection was performed, and stopwords were not removed.

Table 2 shows the ten-fold cross-validation results using the two classifiers. Since all three data sets have an equal distribution between true and false statements, the baseline for all the topics is 50%. The average classification performance of 70% – significantly higher than the 50% baseline – indicates that good separation can be obtained

between true and deceptive language by using automatic classifiers.

Topic	NB	SVM
ABORTION	70.0%	67.5%
DEATH PENALTY	67.4%	65.9%
BEST FRIEND	75.0%	77.0%
AVERAGE	70.8%	70.1%

Table 2: Ten-fold cross-validation classification results, using a Naïve Bayes (NB) or Support Vector Machines (SVM) classifier

To gain further insight into the variation of accuracy with the amount of data available, we also plotted the learning curves for each of the data sets, as shown in Figure 1. The overall growing trend indicates that more data is likely to improve the accuracy, thus suggesting the collection of additional data as a possible step for future work.

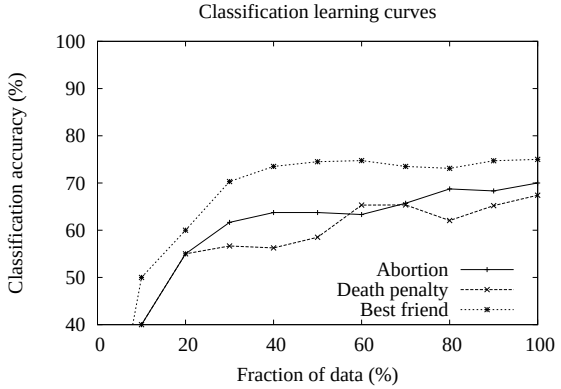


Figure 1: Classification learning curves.

We also tested the portability of the classifiers across topics, using two topics as training data and the third topic as test. The results are shown in Table 3. Although below the in-topic performance, the average accuracy is still significantly higher than the 50% baseline, indicating that the learning process relies on clues specific to truth/deception, and it is not bound to a particular topic.

5 Identifying Dominant Word Classes in Deceptive Text

In order to gain a better understanding of the characteristics of deceptive text, we devised a method to calculate a score associated with a given class of words, as a measure of saliency for the given word class inside the collection of deceptive (or truthful) texts.

Given a class of words $C = \{W_1, W_2, ..., W_N\}$, we define the class coverage in the deceptive corpus D as the percentage of words from D belonging to the class C:

$$Coverage_D(C) = \frac{\sum_{W_i \in C} Frequency_D(W_i)}{Size_D}$$

where $Frequency_D(W_i)$ represents the total number of occurrences of word W_i inside the corpus D, and $Size_D$ represents the total size (in words) of the corpus D.

Similarly, we define the class C coverage for the truthful corpus T:

$$Coverage_T(C) = \frac{\sum_{W_i \in C} Frequency_T(W_i)}{Size_T}$$

The *dominance score* of the class C in the deceptive corpus D is then defined as the ratio between the coverage of the class in the corpus D with respect to the coverage of the same class in the corpus T:

$$Dominance_D(C) = \frac{Coverage_D(C)}{Coverage_T(C)} \quad (1)$$

A dominance score close to 1 indicates a similar distribution of the words in the class C in both the deceptive and the truthful corpus. Instead, a score significantly higher than 1 indicates a class that is dominant in the deceptive corpus, and thus likely to be a characteristic of the texts in this corpus. Finally, a score significantly lower than 1 indicates a class that is dominant in the truthful corpus, and unlikely to appear in the deceptive corpus.

We use the classes of words as defined in the Linguistic Inquiry and Word Count (LIWC), which was developed as a resource for psycholinguistic analysis (Pennebaker and Francis, 1999). The 2001 version of LIWC includes about 2,200 words and word stems grouped into about 70 broad categories relevant to psychological processes (e.g., EMOTION, COGNITION). The LIWC lexicon has been validated by showing significant correlation between human ratings of a large number of written texts and the rating obtained through LIWC-based analyses of the same texts.

All the word classes from LIWC are ranked according to the dominance score calculated with formula 1, using a mix of all three data sets to create the D and T corpora. Those classes that have a high score are the classes that are dominant in deceptive text. The classes that have a small score are the classes that are dominant in truthful text and lack from deceptive text. Table 4 shows the top ranked classes along with their dominance score and a few sample words that belong to the given class and also appeared in the deceptive (truthful) texts.

Interestingly, in both truthful and deceptive language, three of the top five dominant classes are related to humans. In deceptive texts however, the

Training	Test	NB	SVM
DEATH PENALTY + BEST FRIEND	ABORTION	62.0%	61.0%
ABORTION + BEST FRIEND	DEATH PENALTY	58.7%	58.7%
ABORTION + DEATH PENALTY	BEST FRIEND	58.7%	53.6%
AVERAGE		59.8%	57.8%

Table 3: Cross-topic classification results

Class	Score	Sample words
Deceptive Text		
METAPH	1.71	god, die, sacred, mercy, sin, dead, hell, soul, lord, sins
YOU	1.53	you, thou
OTHER	1.47	she, her, they, his, them, him, herself, himself, themselves
HUMANS	1.31	person, child, human, baby, man, girl, humans, individual, male, person, adult
CERTAIN	1.24	always, all, very, truly, completely, totally
Truthful Text		
OPTIM	0.57	best, ready, hope, accepts, accept, determined, accepted, won, super
I	0.59	I, myself, mine
FRIENDS	0.63	friend, companion, body
SELF	0.64	our, myself, mine, ours
INSIGHT	0.65	believe, think, know, see, understand, found, thought, feels, admit

Table 4: Dominant word classes in deceptive text, along with sample words.

human-related word classes (YOU, OTHER, HUMANS) represent detachment from the self, as if trying not to have the own self involved in the lies. Instead, the classes of words that are closely connected to the self (I, FRIENDS, SELF) are lacking from deceptive text, being dominant instead in truthful statements, where the speaker is comfortable with identifying herself with the statements she makes.

Also interesting is the fact that words related to certainty (CERTAIN) are more dominant in deceptive texts, which is probably explained by the need of the speaker to explicitly use truth-related words as a means to emphasize the (fake) "truth" and thus hide the lies. Instead, belief-oriented vocabulary (INSIGHT), such as *believe*, *feel*, *think*, is more frequently encountered in truthful statements, where the presence of the real truth does not require truth-related words for emphasis.

6 Conclusions

In this paper, we explored automatic techniques for the recognition of deceptive language in written texts. Through experiments carried out on three data sets, we showed that truthful and lying texts are separable, and this property holds for different data sets. An analysis of classes of salient features indicated some interesting patterns of word usage in deceptive texts, including detachment from the self and vocabulary that emphasizes certainty. In future work, we plan to explore the role played by affect and the possible integration of automatic emotion analysis into the recognition of deceptive language.

References

B. DePaulo, J. Lindsay, B. Malone, L. Muhlenbruck, K. Charlton, and H. Cooper. 2003. Cues to deception. *Psychological Bulletin*, 129(1):74—118.

J. Hirschberg, S. Benus, J. Brenier, F. Enos, S. Friedman, S. Gilman, C. Girand, M. Graciarena, A. Kathol, L. Michaelis, B. Pellom, E. Shriberg, and A. Stolcke. 2005. Distinguishing deceptive from non-deceptive speech. In *Proceedings of INTERSPEECH-2005*, Lisbon, Portugal.

M. Koppel, S. Argamon, and A. Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 4(17):401–412.

R. Mihalcea and C. Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence*, 22(2):126–142.

M. Newman, J. Pennebaker, D. Berry, and J. Richards. 2003. Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin*, 29:665–675.

J. Pennebaker and M. Francis. 1999. Linguistic inquiry and word count: LIWC. Erlbaum Publishers.

R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii.

L. Zhou, J Burgoon, J. Nunamaker, and D. Twitchell. 2004. Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communication. *Group Decision and Negotiation*, 13:81–106.

Generalizing Dependency Features for Opinion Mining

Mahesh Joshi[1] and **Carolyn Penstein-Rosé**[1,2]
[1]Language Technologies Institute
[2]Human-Computer Interaction Institute
Carnegie Mellon University, Pittsburgh, PA, USA
{maheshj,cprose}@cs.cmu.edu

Abstract

We explore how features based on syntactic dependency relations can be utilized to improve performance on opinion mining. Using a transformation of dependency relation triples, we convert them into "composite back-off features" that generalize better than the regular lexicalized dependency relation features. Experiments comparing our approach with several other approaches that generalize dependency features or ngrams demonstrate the utility of composite back-off features.

1 Introduction

Online product reviews are a crucial source of opinions about a product, coming from the people who have experienced it first-hand. However, the task of a potential buyer is complicated by the sheer number of reviews posted online for a product of his/her interest. Opinion mining, or sentiment analysis (Pang and Lee, 2008) in product reviews, in part, aims at automatically processing a large number of such product reviews to identify opinionated statements, and to classify them into having either a positive or negative polarity.

One of the most popular techniques used for opinion mining is that of supervised machine learning, for which, many different lexical, syntactic and knowledge-based feature representations have been explored in the literature (Dave et al., 2003; Gamon, 2004; Matsumoto et al., 2005; Ng et al., 2006). However, the use of syntactic features for opinion mining has achieved varied results. In our work, we show that by altering syntactic dependency relation triples in a particular way (namely, "backing off" only the head word in a dependency relation to its part-of-speech tag), they generalize better and yield a significant improvement on the task of identifying opinions from product reviews. In effect, this work demonstrates a better way to utilize syntactic dependency relations for opinion mining.

In the remainder of the paper, we first discuss related work. We then motivate our approach and describe the composite back-off features, followed by experimental results, discussion and future directions for our work.

2 Related Work

The use of syntactic or deep linguistic features for opinion mining has yielded mixed results in the literature so far. On the positive side, Gamon (2004) found that the use of deep linguistic features extracted from phrase structure trees (which include syntactic dependency relations) yield significant improvements on the task of predicting satisfaction ratings in customer feedback data. Matsumoto et al. (2005) show that when using frequently occurring sub-trees obtained from dependency relation parse trees as features for machine learning, significant improvement in performance is obtained on the task of classifying movie reviews as having positive or negative polarity. Finally, Wilson et al. (2004) use several different features extracted from dependency parse trees to improve performance on the task of predicting the strength of opinion phrases.

On the flip side, Dave et al. (2003) found that for the task of polarity prediction, adding *adjective-noun* dependency relationships as features does not provide any benefit over a simple bag-of-words based feature space. Ng et al. (2006) proposed that rather than focusing on just *adjective-noun* relationships, the *subject-verb* and *verb-object* relationships should also be considered for polarity classification. However, they observed that the addition of these dependency relationships does not improve performance over a feature space that includes unigrams, bigrams and trigrams.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 313–316,
Suntec, Singapore, 4 August 2009.

One difference that seems to separate the successes from the failures is that of using the entire set of dependency relations obtained from a dependency parser and allowing the learning algorithm to generalize, rather than picking a small subset of dependency relations manually. However, in such a situation, one critical issue might be the sparseness of the very specific linguistic features, which may cause the classifier learned from such features to not generalize. Features based on dependency relations provide a nice way to enable generalization to the right extent through utilization of their structural aspect. In the next section, we motivate this idea in the context of our task, from a linguistic as well as machine learning perspective.

3 Identifying Opinionated Sentences

We focus on the problem of automatically identifying whether a sentence in a product review contains an opinion about the product or one of its features. We use the definition of this task as formulated by Hu and Liu (2004) on Amazon.com and CNet.com product reviews for five different products. Their definition of an opinion sentence is reproduced here verbatim: *"If a sentence contains one or more product features and one or more opinion words, then the sentence is called an opinion sentence."* Any other sentence in a review that does not fit the above definition of an opinion sentence is considered as a non-opinion sentence. In general, these can be expected to be verifiable statements or facts such as product specifications and so on.

Before motivating the use of dependency relations as features for our task, a brief overview about dependency relations follows.

3.1 Dependency Relations

The dependency parse for a given sentence is essentially a set of triplets or triples, each of which is composed of a grammatical relation and the pair of words from the sentence among which the grammatical relation holds ($\{rel_i, w_j, w_k\}$, where rel_i is the dependency relation among words w_j and w_k). The set of dependency relations is specific to a given parser – we use the Stanford parser[1] for computing dependency relations. The word w_j is usually referred to as the *head word* in the dependency triple, and the word w_k is usually referred to as the *modifier word*.

One straightforward way to use dependency relations as features for machine learning is to generate features of the form `RELATION_HEAD_MODIFIER` and use them in a standard bag-of-words type binary or frequency-based representation. The indices of the head and modifier words are dropped for the obvious reason that one does not expect them to generalize across sentences. We refer to such features as *lexicalized dependency relation features*.

3.2 Motivation for our Approach

Consider the following examples (these are made-up examples for the purpose of keeping the discussion succinct, but still capture the essence of our approach):

(i) *This is a great camera!*

(ii) *Despite its few negligible flaws, this really great mp3 player won my vote.*

Both of these sentences have an adjectival modifier (`amod`) relationship, the first one having `amod_camera_great`) and the second one having `amod_player_great`). Although both of these features are good indicators of opinion sentences and are closely related, any machine learning algorithm that treats these features independently will not be able to generalize their relationship to the opinion class. Also, any new test sentence that contains a noun different from either "camera" or "player" (for instance in the review of a different electronic product), but is participating in a similar relationship, will not receive any importance in favor of the opinion class – the machine learning algorithm may not have even seen it in the training data.

Now consider the case where we "back off" the head word in each of the above features to its part-of-speech tag. This leads to a single feature: `amod_NN_great`. This has two advantages: first, the learning algorithm can now learn a weight for a more general feature that has stronger evidence of association with the opinion class, and second, any new test sentence that contains an unseen noun in a similar relationship with the adjective "great" will receive some weight in favor of the opinion class. This "back off" operation is a generalization of the regular lexicalized dependency relations mentioned above. In the next section we describe all such generalizations that we experimented with.

[1] `http://nlp.stanford.edu/software/lex-parser.shtml`

4 Methodology

Composite Back-off Features: The idea behind our composite back-off features is to create more generalizable, but not overly general back-off features by backing off to the part-of-speech (POS) tag of either the head word or the modifier word (but not both at once, as in Gamon (2004) and Wilson et al. (2004)) – hence the description "composite," as there is a lexical part to the feature, coming from one word, and a POS tag coming from the other word, along with the dependency relation itself.

The two types of composite back-off features that we create from lexicalized dependency triples are as follows:

(i) **h-bo**: Here we use features of the form $\{rel_i, POS_j, w_k\}$ where the head word is replaced by its POS tag, but the modifier word is retained.

(ii) **m-bo**: Here we use features of the form $\{rel_i, w_j, POS_k\}$, where the modifier word is replaced by its POS tag, but the head word is retained.

Our hypothesis is that the **h-bo** features will perform better than purely lexicalized dependency relations for reasons mentioned in Section 3.2 above. Although **m-bo** features also generalize the lexicalized dependency features, in a relation such as an adjectival modifier (discussed in Section 3.2 above), the head noun is a better candidate to back-off for enabling generalization across different products, rather than the modifier adjective. For this reason, we do not expect their performance to be comparable to **h-bo** features.

We compare our composite back-off features with other similar ways of generalizing dependency relations and lexical ngrams that have been tried in previous work. We describe these below.

Full Back-off Features: Both Gamon (2004) and Wilson et al. (2004) utilize features based on the following version of dependency relationships: $\{rel_i, POS_j, POS_k\}$, where they "back off" both the head word and the modifier word to their respective POS tags (POS_j and POS_k). We refer to this as **hm-bo**.

NGram Back-off Features: Similar to McDonald et al. (2007), we utilize backed-off versions of lexical bigrams and trigrams, where all possible combinations of the words in the ngram are replaced by their POS tags, creating features such as $w_j\_POS_k$, $POS_j\_w_k$, $POS_j\_POS_k$ for each lexical bigram and similarly for trigrams. We refer to these as **bi-bo** and **tri-bo** features respectively.

In addition to these back-off approaches, we also use regular lexical bigrams (**bi**), lexical trigrams (**tri**), POS bigrams (**POS-bi**), POS trigrams (**POS-tri**) and lexicalized dependency relations (**lexdep**) as features. While testing all of our feature sets, we evaluate each of them individually by adding them to the basic set of unigram (**uni**) features.

5 Experiments and Results

Details of our experiments and results follow.

5.1 Dataset

We use the extended version of the Amazon.com / CNet.com product reviews dataset released by Hu and Liu (2004), available from their web page[2]. We use a randomly chosen subset consisting of 2,200 review sentences (200 sentences each for 11 different products)[3]. The distribution is 1,053 (47.86%) opinion sentences and 1,147 (52.14%) non-opinion sentences.

5.2 Machine Learning Parameters

We have used the Support Vector Machine (SVM) learner (Shawe-Taylor and Cristianini, 2000) from the MinorThird Toolkit (Cohen, 2004), along with the χ-squared feature selection procedure, where we reject features if their χ-squared score is not significant at the 0.05 level. For SVM, we use the default linear kernel with all other parameters also set to defaults. We perform 11-fold cross-validation, where each test fold contains all the sentences for one of the 11 products, and the sentences for the remaining ten products are in the corresponding training fold. Our results are reported in terms of average accuracy and Cohen's kappa values across the 11 folds.

5.3 Results

Table 1 shows the full set of results from our experiments. Our results are comparable to those reported by Hu and Liu (2004) on the same task; as well as those by Arora et al. (2009) on a similar task of identifying qualified vs. bald claims in product reviews. On the accuracy metric, the composite features with the head word backed off

[2] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

[3] http://www.cs.cmu.edu/~maheshj/datasets/acl09short.html

Features	Accuracy	Kappa
uni	.652 (±.048)	.295 (±.049)
uni+bi	.657 (±.066)	.304 (±.089)
uni+bi-bo	.650 (±.056)	.299 (±.079)
uni+tri	.655 (±.062)	.306 (±.077)
uni+tri-bo	.647 (±.051)	.287 (±.075)
uni+POS-bi	.676 (±.057)	**.349 (±.083)**
uni+POS-tri	.661 (±.050)	.317 (±.064)
uni+lexdep	.639 (±.055)	.268 (±.079)
uni+hm-bo	.670 (±.046)	.336 (±.065)
uni+h-bo	**.679 (±.063)**	**.351 (±.097)**
uni+m-bo	.657 (±.056)	.308 (±.063)

Table 1: Shown are the average accuracy and Cohen's kappa across 11 folds. Bold indicates statistically significant improvements ($p < 0.05$, two-tailed pairwise T-test) over the (**uni**) baseline.

are the only ones that achieve a statistically significant improvement over the **uni** baseline. On the kappa metric, using POS bigrams also achieves a statistically significant improvement, as do the composite **h-bo** features. None of the other back-off strategies achieve a statistically significant improvement over **uni**, although numerically **hm-bo** comes quite close to **h-bo**. Evaluation of these two types of features by themselves (without unigrams) shows that **h-bo** are significantly better than **hm-bo** at $p < 0.10$ level. Regular lexicalized dependency relation features perform worse than unigrams alone. These results thus demonstrate that composite back-off features based on dependency relations, where only the head word is backed off to its POS tag present a useful alternative to encoding dependency relations as features for opinion mining.

6 Conclusions and Future Directions

We have shown that for opinion mining in product review data, a feature representation based on a simple transformation ("backing off" the head word in a dependency relation to its POS tag) of syntactic dependency relations captures more generalizable and useful patterns in data than purely lexicalized dependency relations, yielding a statistically significant improvement.

The next steps that we are currently working on include applying this approach to polarity classification. Also, the aspect of generalizing features across different products is closely related to fully supervised domain adaptation (Daumé III, 2007), and we plan to combine our approach with the idea from Daumé III (2007) to gain insights into whether the composite back-off features exhibit different behavior in domain-general versus domain-specific feature sub-spaces.

Acknowledgments

This research is supported by National Science Foundation grant IIS-0803482.

References

Shilpa Arora, Mahesh Joshi, and Carolyn Rosé. 2009. Identifying Types of Claims in Online Customer Reviews. In *Proceedings of NAACL 2009*.

William Cohen. 2004. Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data.

Hal Daumé III. 2007. Frustratingly Easy Domain Adaptation. In *Proceedings of ACL 2007*.

Kushal Dave, Steve Lawrence, and David Pennock. 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In *Proceedings of WWW 2003*.

Michael Gamon. 2004. Sentiment Classification on Customer Feedback Data: Noisy Data, Large Feature Vectors, and the Role of Linguistic Analysis. In *Proceedings of COLING 2004*.

Minqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *Proceedings of ACM SIGKDD 2004*.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees. In *Proceedings of the 9th PAKDD*.

Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. 2007. Structured Models for Fine-to-Coarse Sentiment Analysis. In *Proceedings of ACL 2007*.

Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. 2006. Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews. In *Proceedings of the COLING/ACL 2006*.

Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1–2).

John Shawe-Taylor and Nello Cristianini. 2000. *Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. 2004. Just How Mad Are You? Finding Strong and Weak Opinion Clauses. In *Proceedings of AAAI 2004*.

Graph Ranking for Sentiment Transfer

Qiong Wu[1,2], Songbo Tan[1] and Xueqi Cheng[1]
[1]Institute of Computing Technology, Chinese Academy of Sciences, China
[2] Graduate University of Chinese Academy of Sciences, China
{wuqiong,tansongbo}@software.ict.ac.cn, cxq@ict.ac.cn

Abstract

With the aim to deal with sentiment-transfer problem, we proposed a novel approach, which integrates the sentiment orientations of documents into the graph-ranking algorithm. We apply the graph-ranking algorithm using the accurate labels of old-domain documents as well as the "pseudo" labels of new-domain documents. Experimental results show that proposed algorithm could improve the performance of baseline methods dramatically for sentiment transfer.

1 Introduction

With the rapid growth of reviewing pages, sentiment classification is drawing more and more attention (Bai et al., 2005; Pang and Lee, 2008). Generally speaking, sentiment classification can be considered as a special kind of traditional text classification (Tan et al., 2005; Tan, 2006). In most cases, supervised learning methods can perform well (Pang et al., 2002). But when training data and test data are drawn from different domains, supervised learning methods always produce disappointing results. This is so-called cross-domain sentiment classification problem (or sentiment-transfer problem).

Sentiment transfer is a new study field. In recent years, only a few works are conducted on this field. They are generally divided into two categories. The first one needs a small amount of labeled training data for the new domain (Aue and Gamon, 2005). The second one needs no labeled data for the new domain (Blitzer et al., 2007; Tan et al., 2007; Andreevskaia and Bergler, 2008; Tan et al., 2008; Tan et al., 2009). In this paper, we concentrate on the second category which proves to be used more widely.

Graph-ranking algorithm has been successfully used in many fields (Wan et al., 2006; Esuli and Sebastiani, 2007), whose idea is to give a node high score if it is strongly linked with other high-score nodes. In this work, we extend the graph-ranking algorithm for sentiment transfer by integrating the sentiment orientations of the documents, which could be considered as a sentiment-transfer version of the graph-ranking algorithm. In this algorithm, we assign a score for every unlabelled document to denote its extent to "negative" or "positive", then we iteratively calculate the score by making use of the accurate labels of old-domain data as well as the "pseudo" labels of new-domain data, and the final score for sentiment classification is achieved when the algorithm converges, so we can label the new-domain data based on these scores.

2 The Proposed Approach

2.1 Overview

In this paper, we have two document sets: the test data $D^U = \{d_1,...,d_n\}$ where d_i is the term vector of the i^{th} text document and each $d_i \in D^U (i = 1,\ldots,n)$ is unlabeled; the training data $D^L = \{d_{n+1},...d_{n+m}\}$ where d_j represents the term vector of the j^{th} text document and each $d_j \in D^L (j = n+1,\ldots,n+m)$ should have a label from a category set C = {negative, positive}. We assume the training dataset D^L is from the related but different domain with the test dataset D^U. Our objective is to maximize the accuracy of assigning a label in C to $d_i \in D^U$ $(i = 1,\ldots,n)$ utilizing the training data D^L in another domain.

The proposed algorithm is based on the following presumptions:

(1) Let W^L denote the word space of old domain, W^U denote the word space of new domain. $W^L \cap W^U \neq \Phi$.

(2) The labels of documents appear both in the training data and the test data should be the same.

Based on graph-ranking algorithm, it is thought that if a document is strongly linked with positive (negative) documents, it is probably positive (negative). And this is the basic idea of learning from a document's neighbors.

Our algorithm integrates the sentiment orientations of the documents into the graph-ranking algorithm. In our algorithm, we build a graph

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 317–320,
Suntec, Singapore, 4 August 2009.

whose nodes denote documents and edges denote the content similarities between documents. We initialize every document a score ("1" denotes positive, and "-1" denotes negative) to represent its degree of sentiment orientation, and we call it sentiment score. The proposed algorithm calculates the sentiment score of every unlabelled document by learning from its neighbors in both old domain and new domain, and then iteratively calculates the scores with a unified formula. Finally, the algorithm converges and each document gets its sentiment score. When its sentiment score falls in the range [0, 1] (or [-1, 0]], the document should be classified as "positive (or negative)". The closer its sentiment score is near 1 (or -1), the higher the "positive (or negative)" degree is.

2.2 Score Documents

Score Documents Using Old-domain Information

We build a graph whose nodes denote documents in both D^L and D^U and edges denote the content similarities between documents. If the content similarity between two documents is 0, there is no edge between the two nodes. Otherwise, there is an edge between the two nodes whose weight is the content similarity. The content similarity between two documents is computed with the cosine measure. We use an adjacency matrix U to denote the similarity matrix between D^U and D^L. $U=[U_{ij}]_{nxm}$ is defined as follows:

$$U_{ij} = \frac{d_i \bullet d_j}{\|d_i\| \times \|d_j\|}, \quad i = 1,...,n, j = n+1,...,n+m \qquad (1)$$

The weight associated with term t is computed with $tf_t idf_t$ where tf_t is the frequency of term t in the document and idf_t is the inverse document frequency of term t, i.e. $1+\log(N/n_t)$, where N is the total number of documents and n_t is the number of documents containing term t in a data set.

In consideration of convergence, we normalize U to $\hat{U}$ by making the sum of each row equal to 1:

$$\hat{U}_{ij} = \begin{cases} U_{ij} \Big/ \sum_{j=1}^{m} U_{ij}, & if \sum_{j=1}^{m} U_{ij} \neq 0 \\ 0, & otherwise \end{cases} \qquad (2)$$

In order to find the neighbors (in another word, the nearest documents) of a document, we sort every row of $\hat{U}$ to $\tilde{U}$ in descending order. That is: $\tilde{U}_{ij} \geq \tilde{U}_{ik}$ $(i = 1,\ldots n; j,k = 1,\ldots m; k \geqslant j)$.

Then for $d_i \in D^U$ $(i = 1,\ldots,n)$, $\tilde{U}_{ij}$ $(j = 1,\ldots,K)$ corresponds to K neighbors in D^L. So we can get its K neighbors. We use a matrix $N = [N_{ij}]_{n \times K}$ to denote the neighbors of D^U in old domain, with N_{ij} corresponding to the j^{th} nearest neighbor of d_i.

At last, we can calculate sentiment score s_i $(i = 1,\ldots,n)$ using the scores of the d_i's neighbors as follows:

$$s_i^{(k)} = \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}), \quad i = 1,...,n \qquad (3)$$

where $i \bullet$ means the i^{th} row of a matrix and $s_i^{(k)}$ denotes the s_i at the k^{th} iteration.

Score Documents Using New-domain Information

Similarly, a graph is built, in which each node corresponds to a document in D^U and the weight of the edge between any different documents is computed by the cosine measure. We use an adjacency matrix $V=[V_{ij}]_{n \times n}$ to describe the similarity matrix. And V is similarly normalized to $\hat{V}$ to make the sum of each row equal to 1. Then we sort every row of $\hat{V}$ to $\tilde{V}$ in descending order, thus we can get K neighbors of $d_i \in D^U$ $(i = 1,\ldots,n)$ from $\tilde{V}_{ij}$ $(j = 1,\ldots K)$, and we use a matrix $M = [M_{ij}]_{n \times K}$ to denote the neighbors of D^U in the new domain. Finally, we can calculate s_i using the sentiment scores of the d_i's neighbors as follows:

$$s_i^{(k)} = \sum_{j \in M_{i\bullet}} (\hat{V}_{ij} \times s_j^{(k-1)}), \quad i = 1,...,n \qquad (4)$$

2.3 Sentiment Transfer Algorithm

Initialization

Firstly, we classify the test data D^U to get their initial labels using a traditional classifier. For simplicity, we use prototype classification algorithm (Tan et al., 2005) in this work.

Then, we give "-1" to $s_i^{(0)}$ if d_i's label is "negative", and "1" if "positive". So we obtain the initial sentiment score vector $S^{(0)}$ for both domain data.

At last, $s_i^{(0)}$ $(i = 1,\ldots,n)$ is normalized as follows to make the sum of positive scores of D^U equal to 1, and the sum of negative scores of D^U equal to -1:

$$s_i^{(0)} = \begin{cases} s_i^{(0)} \Big/ \sum_{j \in D_{neg}^U} (-s_j^{(0)}), & if \quad s_i^{(0)} < 0 \\ s_i^{(0)} \Big/ \sum_{j \in D_{pos}^U} s_j^{(0)}, & if \quad s_i^{(0)} > 0 \end{cases} \quad i = 1,...,n \qquad (5)$$

where D^U_{neg} and D^U_{pos} denote the negative and positive document set of D^U respectively. The same as (5), $s_j^{(0)}$ (j =n+1,…,n+m) is normalized.

Algorithm Introduction

In our algorithm, we label D^U by making use of information of both old domain and new domain. We fuse equations (3) and (4), and get the iterative equation as follows:

$$s_i^{(k)} = \alpha \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}) + \beta \sum_{h \in M_{i\bullet}} (\hat{V}_{ih} \times s_h^{(k-1)}), i = 1,...,n \quad (6)$$

where $\alpha + \beta = 1$, and α and β show the relative importance of old domain and new domain to the final sentiment scores. In consideration of the convergence, $S^{(k)}$ (S at the k^{th} iteration) is normalized after each iteration.

Here is the complete algorithm:

1. Classify D^U with a traditional classifier. Initialize the sentiment score s_i of $d_i \in D^U \cup D^L$ (i = 1,…n+m) and normalize it.
2. Iteratively calculate the $S^{(k)}$ of D^U and normalize it until it achieves the convergence:

$$s_i^{(k)} = \alpha \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}) + \beta \sum_{h \in M_{i\bullet}} (\hat{V}_{ih} \times s_h^{(k-1)}), i = 1,...,n$$

$$s_i^{(k)} = \begin{cases} s_i^{(k)} \Big/ \sum_{j \in D^U_{neg}} (-s_j^{(k)}), & if \quad s_i^{(k)} < 0 \\ s_i^{(k)} \Big/ \sum_{j \in D^U_{pos}} s_j^{(k)}, & if \quad s_i^{(k)} > 0 \end{cases} \quad i = 1,...,n$$

3. According to $s_i \in S$ (i = 1,…,n), assign each $d_i \in D^U$ (i = 1,…n) a label. If s_i is between -1 and 0, assign d_i the label "negative"; if s_i is between 0 and 1, assign d_i the label "positive".

3 EXPERIMENTS

3.1 Data Preparation

We prepare three Chinese domain-specific data sets from on-line reviews, which are: Electronics Reviews (Elec, from http://detail.zol.com.cn/), Stock Reviews (Stock, from http://blog.sohu.com /stock/) and Hotel Reviews (Hotel, from http://www.ctrip.com/). And then we manually label the reviews as "negative" or "positive".

The detailed composition of the data sets are shown in Table 1, which shows the name of the data set (DataSet), the number of negative reviews (Neg), the number of positive reviews (Pos), the average length of reviews (Length), the number of different words (Vocabulary) in this data set.

DataSet	Neg	Pos	Length	Vocabulary
Elec	554	1,054	121	6,200
Stock	683	364	460	13,012
Hotel	2,000	2,000	181	11,336

Table 1. Data sets composition

We make some preprocessing on the datasets. First, we use ICTCLAS (http://ictclas.org/), a Chinese text POS tool, to segment these Chinese reviews. Second, the documents are represented by vector space model.

3.2 Evaluation Setup

In our experiment, we use prototype classification algorithm (Tan et al., 2005) and Support Vector Machine experimenting on the three data sets as our baselines separately. The Support Vector Machine is a state-of-the-art supervised learning algorithm. In our experiment, we use LibSVM (www.csie.ntu.edu.tw/~cjlin/libsvm/) with a linear kernel and set all options by default.

We also compare our algorithm to Structural Correspondence Learning (SCL) (Blitzer et al., 2007). SCL is a state-of-the-art sentiment-transfer algorithm which automatically induces correspondences among features from different domains. It identifies correspondences among features from different domains by modeling their correlations with pivot features, which are features that behave in the same way for discriminative learning in both domains. In our experiment, we use 100 pivot features.

3.3 Overall Performance

In this section, we conduct two groups of experiments where we separately initialize the sentiment scores in our algorithm by prototype classifier and Support Vector Machine.

There are two parameters in our algorithm, K and α (β can be calculated by 1-α). We set the parameters K and α with 150 and 0.7 respectively, which indicates we use 150 neighbors and the contribution from old domain is a little more important than that from new domain. It is thought that the algorithm achieves the convergence when the changing between the sentiment score s_i computed at two successive iterations for any $d_i \in D^U$ (i = 1,…n) falls below a given threshold, and we set the threshold 0.00001 in this work.

Table 2 shows the accuracy of Prototype, LibSVM, SCL and our algorithm when training data and test data belong to different domains.

Our algorithm is separately initialized by Prototype and LibSVM.

	Baseline		SCL	Proposed Algorithm	
	Prototype	LibSVM		Prototype+ OurApproach	LibSVM+ OurApproach
Elec->Stock	0.6652	0.6478	**0.7507**	0.7326	0.7304
Elec->Hotel	0.7304	0.7522	**0.7750**	0.7543	0.7543
Stock->Hotel	0.6848	0.6957	**0.7683**	0.7435	0.7457
Stock->Elec	0.7043	0.6696	0.8340	**0.8457**	0.8435
Hotel->Stock	0.6196	0.5978	0.6571	**0.7848**	**0.7848**
Hotel->Elec	0.6674	0.6413	0.7270	**0.8609**	**0.8609**
Average	0.6786	0.6674	0.7520	**0.7870**	0.7866

Table 2. Accuracy comparison of different methods

As we can observe from Table 2, our algorithm can dramatically increase the accuracy of sentiment-transfer. Seen from the 2nd column and the 5th column, every accuracy of the proposed algorithm is increased comparing to Prototype. The average increase of accuracy over all the 6 problems is 10.8%. Similarly, the accuracy of our algorithm is higher than LibSVM in every problem and the average increase of accuracy is 11.9%. The great improvement comparing with the baselines indicates that the proposed algorithm performs very effectively and robustly.

Seen from Table 2, our result about SCL is in accord with that in (Blitzer et al., 2007) on the whole. The average accuracy of SCL is higher than both baselines, which convinces that SCL is effective for sentiment-transfer. However, our approach outperforms SCL: the average accuracy of our algorithm is about 3.5 % higher than SCL. This is caused by two reasons. First, SCL is essentially based on co-occurrence of words (the window size is the whole document), so it is easily affected by low frequency words and the size of data set. Second, the pivot features of SCL are totally dependent on experts in the field, so the quality of pivot features will seriously affect the performance of SCL. This improvement convinces us of the effectiveness of our algorithm.

4 Conclusion and Future Work

In this paper, we propose a novel sentiment-transfer algorithm. It integrates the sentiment orientations of the documents into the graph-ranking based method for sentiment-transfer problem. The algorithm assigns a score for every document being predicted, and it iteratively calculates the score making use of the accurate labels of old-domain data, as well as the "pseudo" labels of new-domain data, finally it labels the new-domain data as "negative" or "positive" basing on this score. The experiment results show that the proposed approach can dramatically improve the accuracy when transferred to a new domain.

In this study, we find the neighbors of a given document using cosine similarity. This is too general, and perhaps not so proper for sentiment classification. In the next step, we will try other methods to calculate the similarity. Also, our approach can be applied to multi-task learning.

5 Acknowledgments

This work was mainly supported by two funds, i.e., 0704021000 and 60803085, and one another project, i.e., 2004CB318109.

References

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2008

S. Tan, X. Cheng, M. Ghanem, B. Wang and H. Xu. 2005. A Novel Refinement Approach for Text Categorization. In *Proceedings of CIKM 2005*.

S. Tan. 2006. An Effective Refinement Strategy for KNN Text Classifier. Expert Systems With Applications. Elsevier. 30(2): 290-298.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP, 2002*.

X. Bai, R. Padman and E. Airoldi. 2005. On learning parsimonious models for extracting consumer opinions. In *Proceedings of HICSS 2005*.

A. Aue and M. Gamon. 2005. Customizing sentiment classifiers to new domains: a case study. In *Proceedings of RANLP 2005*.

J. Blitzer, M. Dredze, and F. Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL 2007*.

S. Tan, G. Wu, H. Tang and X. Cheng. 2007. A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceedings of CIKM 2007*.

S. Tan, Y. Wang, G. Wu and X. Cheng. 2008. Using unlabeled data to handle domain-transfer problem of semantic detection. In *Proceedings of SAC 2008*.

S. Tan, X. Cheng, Y. Wang, H. Xu. 2009. Adapting Naive Bayes to Domain Adaptation for Sentiment Analysis. In *Proceedings of ECIR 2009*.

A. Esuli, F. Sebastiani. 2007. Random-walk models of term semantics: An application to opinion-related properties. In *Proceedings of LTC 2007*.

X. Wan, J. Yang and J. Xiao. 2006. Using Cross-Document Random Walks for Topic-Focused Multi-Document Summarization. In *Proceedings of WI 2006*.

A. Andreevskaia and S. Bergler. 2008. When Specialists and Generalists Work Together: Overcoming Domain Dependence in Sentiment Tagging. In *Proceedings of ACL 2008*.

The Contribution of Stylistic Information to Content-based Mobile Spam Filtering

Dae-Neung Sohn and **Jung-Tae Lee** and **Hae-Chang Rim**
Department of Computer and Radio Communications Engineering
Korea University
Seoul, 136-713, South Korea
{danny,jtlee,rim}@nlp.korea.ac.kr

Abstract

Content-based approaches to detecting mobile spam to date have focused mainly on analyzing the *topical* aspect of a SMS message (*what* it is about) but not on the *stylistic* aspect (*how* it is written). In this paper, as a preliminary step, we investigate the utility of commonly used stylistic features based on shallow linguistic analysis for learning mobile spam filters. Experimental results show that the use of stylistic information is potentially effective for enhancing the performance of the mobile spam filters.

1 Introduction

Mobile spam, also known as SMS spam, is a subset of spam that involves unsolicited advertising text messages sent to mobile phones through the Short Message Service (SMS) and has increasingly become a major issue from the early 2000s with the popularity of mobile phones. Governments and many service providers have taken various countermeasures in order to reduce the number of mobile spam (*e.g.* by imposing substantial fines on spammers, blocking specific phone numbers, creating an alias address, *etc.*). Nevertheless, the rate of mobile spam continues to rise.

Recently, a more technical approach to mobile spam filtering based on the content of a SMS message has started gaining attention in the spam research community. Gómez Hidalgo et al. (2006) previously explored the use of statistical learning-based classifiers trained with lexical features, such as character and word n-grams, for mobile spam filtering. However, content-based spam filtering directed at SMS messages are very challenging, due to the fact that such messages consist of only a few words. More recent studies focused on expanding the feature set for learning-based mobile spam classifiers with additional features, such as orthogonal sparse word bi-grams (Cormack et al., 2007a; Cormack et al., 2007b).

Collectively, the features exploited in earlier content-based approach to mobile spam filtering are topical terms or phrases that statistically indicate the spamness of a SMS message, such as *"loan"* or *"70% off sale"*. However, there is no guarantee that legitimate (non-spam) messages would not contain such expressions. Any of us may send a SMS message such as *"need ur advise on private **loans**, plz call me"* or *"mary, abc.com is having **70% off sale** today"*. For current content-based mobile spam filters, there is a chance that they would classify such legitimate messages as spam. This motivated us to not only rely on the message content itself but incorporate new features that reflect its "style," the manner in which the content is expressed, in mobile spam filtering.

The main goal of this paper is to investigate the potential of stylistic features in improving the performance of learning-based mobile spam filters. In particular, we adopt stylistic features previously suggested in authorship attribution studies based on stylometry, the statistical analysis of linguistic style.[1] Our assumption behind adopting the features from authorship attribution are as follows:

- There are two types of SMS message senders, namely *spammers* and *non-spammers*.
- Spammers have distinctive linguistic styles and writing behaviors (as opposed to non-spammers) and use them consistently.
- The SMS message as an end product carries the author's "fingerprints".

[1] Authorship attribution involves identifying the author of a text given some stylistic characteristics of authors' writing. See Holmes (1998) for overview.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 321–324,
Suntec, Singapore, 4 August 2009.

Although there are many types of stylistic features suggested in the literature, we make use of the ones that are readily computable and countable from SMS message texts without any complex linguistic analysis as a preliminary step, including word and sentence lengths (Mendenhall, 1887), frequencies of function words (Mosteller and Wallace, 1964), and part-of-speech tags and tag n-grams (Argamon-Engelson et al., 1998; Koppel et al., 2003; Santini, 2004).

Our experimental result on a large-scale, real world SMS dataset demonstrates that the newly added stylistic features effectively contributes to statistically significant improvement on the performance of learning-based mobile spam filters.

2 Stylistic Feature Set

All stylistic features listed below have been automatically extracted using shallow linguistic analysis. Note that most of them have been motivated from previous stylometry studies.

2.1 Length features: *LEN*

Mendenhall (1887) first created the idea of counting word lengths to judge the authorship of texts, followed by Yule (1939) and Morton (1965) with the use of sentence lengths. In this paper, we measure the overall byte length of SMS messages and the average byte length of words in the message as features.

2.2 Function word frequencies: *FW*

Motivated from a number of stylometry studies based on function words including Mosteller and Wallace (1964), Tweedie et al. (1996) and Argamon and Levitan (2005), we measure the frequencies of function words in SMS messages as features. The intuition behind function words is that due to their high frequency in languages and highly grammaticalized roles, such words are unlikely to be subject to conscious control by the author and that the frequencies of different function words would vary greatly across different authors (Argamon and Levitan, 2005).

2.3 Part-of-speech n-grams: *POS*

Following the work of Argamon-Engelson et al. (1998), Koppel et al. (2003), Santini (2004) and Gamon (2004), we extract part-of-speech n-grams (up to trigrams) from the SMS messages and use their frequencies as features. The idea behind their utility is that spammers would favor certain syntactic constructions in their messages.

2.4 Special characters: *SC*

We have observed that many SMS messages contain special characters and that their usage varies between spam and non-spam messages. For instance, non-spammers often use special characters to create emoticons to express their mood, such as ":-)" (smiling) or "T_T" (crying), whereas spammers tend to use special character or patterns related to monetary matters, such as "$$$" or "%". Therefore, we also measured the ratio of special characters, the number of emoticons, and the number of special character patterns in SMS messages as features.[2]

3 Learning a Mobile Spam Filter

In this paper, we use maximum entropy model, which have shown robust performance in various text classification tasks in the literature, for learning the mobile spam filter. Simply put, given a number of training samples (in our case, SMS messages), each with a label Y (where $Y = 1$ if spam and 0 otherwise) and a feature vector $\overline{x}$, the filter learns a vector of feature weight parameters $\overline{w}$. Given a test sample X with its feature vector $\overline{x}$, the filer outputs the conditional probability of predicting the data as spam, $P(Y = 1|X = \overline{x})$. We use the L-BFGS algorithm (Malouf, 2002) and the Information Gain (IG) measure for parameter estimation and feature selection, respectively.

4 Experiments

4.1 SMS test collections

We use a collection of mobile SMS messages in Korean, with 18,000 (60%) legitimate messages and 12,000 (40%) spam messages. This collection is based on one used in our previous work (Sohn et al., 2008) augmented with 10,000 new messages. Note that the size is approximately 30 times larger than the most previous work by Cormack et al. (2007a) on mobile spam filtering.

4.2 Feature setting

We compare three types of feature sets, as follows:

[2]For emoticon and special pattern counts, we used manually constructed lexicons consisting of 439 emoticons and 229 special patterns.

- *Baseline*: This set consists of lexical features in SMS messages, including words, character n-grams, and orthogonal sparse word bigrams (OSB)[3]. This feature set represents the content-based approaches previously proposed by Gómez Hidalgo et al. (2006), Cormack et al. (2007a) and Cormack et al. (2007b).

- *Proposed*: This feature set consists of all the stylistic features mentioned in Section 2.

- *Combined*: This set is a combination of both the baseline and proposed feature sets.

For all three sets, we make use of 100 features with the highest IG values.

4.3 Evaluation measures

Since spam filtering task is very sensitive to false-positives (*i.e.* legitimate classified as spam) and false-negatives (*i.e.* spam classified as legitimate), special care must be taken when choosing an appropriate evaluation criterion.

Following the TREC Spam Track, we evaluate the filters using ROC curves that plot false-positive rate against false-negative rate. As a summary measure, we report one minus area under the ROC curve (1−AUC) as a percentage with confidence intervals, which is the TREC's official evaluation measure.[4] Note that *lower* 1−AUC(%) value means *better* performance. We used the TREC Spam Filter Evaluation Toolkit[5] in order to perform the ROC analysis.

4.4 Results

All experiments were performed using 10-fold cross validation. Statistical significance of differences between results were computed with a two-tailed paired t-test. The symbol † indicates statistical significance over an appropriate baseline at $p < 0.01$ level.

Table 1 reports the 1−AUC(%) summary for each feature settings listed in Section 4.2. Notice that *Proposed* achieves significantly better performance than *Baseline*. (Recall that the smaller, the better.) An even greater performance gain is obtained by combining both *Proposed* and *Baseline*. This clearly indicates that stylistic aspects of SMS messages are potentially effective for mobile spam filtering.

Feature set	1−AUC (%)	
Baseline	10.7227	[9.4476 - 12.1176]
Proposed	**4.8644**†	[4.2726 - 5.5886]
Combined	**3.7538**†	[3.1186 - 4.4802]

Table 1: Performance of different feature settings.

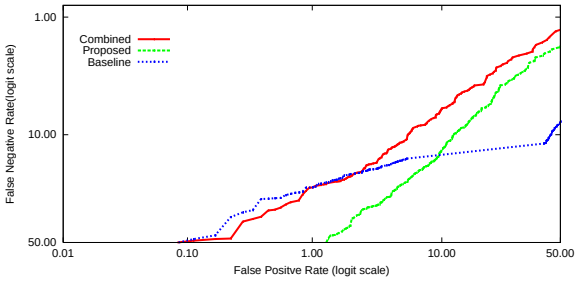


Figure 1: ROC curves of different feature settings.

Figure 1 shows the ROC curves of each feature settings. Notice the tradeoff when *Proposed* is used solely with comparison to *Baseline*; false-positive rate is worsened in return for gaining better false-negative rate. Fortunately, when both feature sets are combined, false-positive rate is remained unchanged while the lowest false-negative rate is achieved. This suggests that the addition of stylistic features contributes to the enhancement of false-negative rate while not hurting false-positive rate (*i.e.* the cases where spam is classified as legitimate are significantly lessened).

In order to evaluate the contribution of different types of stylistic features, we conducted a series of experiments by removing features of a specific type at a time from *Combined*. Table 2 shows the detailed result. Notice that *LEN* and *SC* features are the most helpful, since the performance drops significantly after removing either of them. Interestingly, *FW* and *POS* features show similar contributions; we suggest that these two feature types have similar effects in this filtering task.

We also conducted another series of experiments, by adding one feature type at a time to *Baseline*. Table 3 reports the results. Notice that *LEN* features are consistently the most helpful. The most interesting result is that *POS* features continuously contributes the least. We carefully

[3]OSB refers to words separated by 3 or fewer words, along with an indicator of the difference in word positions; for example, the expression *"the quick brown fox"* would induce following OSB features: *"the* (0) *quick"*, *"the* (1) *brown"*, *"the* (2) *fox"*, *"quick* (0) *brown"*, *"quick* (1) *fox"*, and *"brown* (0) *fox"* (Cormack et al., 2007a).

[4]For detail on ROC analysis, see Cormack et al. (2007a).

[5]Available at http://plg.uwaterloo.ca/.trlynam/spamjig/

Feature set	1−AUC (%)	
Combined	3.7538	[3.1186 - 4.4802]
− LEN	**4.7351**†	[4.0457 - 5.6405]
− FW	3.9823†	[3.3048 - 4.5930]
− POS	4.0712†	[3.4057 - 4.8630]
− SC	**4.7644**†	[4.1012 - 5.4350]

Table 2: Performance by removing one stylistic feature set from the *Combined* set.

Feature set	1−AUC (%)	
Baseline	10.7227	[9.4476 - 12.1176]
+ LEN	**5.5275**†	[4.0457 - 6.6281]
+ FW	6.0828†	[5.1783 - 6.9249]
+ POS	9.6103†	[8.7190 - 11.0579]
+ SC	7.5288†	[6.6049 - 8.4466]

Table 3: Performance by adding one stylistic feature set to the *Baseline* set.

hypothesize that the result is due to high dependencies between *POS* and lexical features.

5 Discussion

In this paper, we have introduced new features that indicate the written style of texts for content-based mobile spam filtering. We have also shown that the stylistic features are potentially useful in improving the performance of mobile spam filters.

This is definitely a work in progress, and much more experimentation is required. Deep linguistic analysis-based stylistic features, such as context free grammar production frequencies (Gamon, 2004) and syntactic rewrite rules in an automatic parse (Baayen et al., 1996), that have already been successfully used in the stylometry literature may be considered. Perhaps most importantly, the method must be tested on various mobile spam data sets written in languages other than Korean. These would be our future work.

References

Shlomo Argamon and Shlomo Levitan. 2005. Measuring the usefulness of function words for authorship attribution. In *Proceedings of ACH/ALLC '05*.

Shlomo Argamon-Engelson, Moshe Koppel, and Galit Avneri. 1998. Style-based text categorization: What newspaper am i reading? In *Proceedings of AAAI '98 Workshop on Text Categorization*, pages 1–4.

H. Baayen, H. van Halteren, and F. Tweedie. 1996. Outside the cave of shadows: using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, 11(3):121–132.

Gordon V. Cormack, José María Gómez Hidalgo, and Enrique Puertas Sánz. 2007a. Spam filtering for short messages. In *Proceedings of CIKM '07*, pages 313–320.

Gordon V. Cormack, José María Gómez Hidalgo, and Enrique Puertas Sánz. 2007b. Feature engineering for mobile (sms) spam filtering. In *Proceedings of SIGIR '07*, pages 871–872.

Michael Gamon. 2004. Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In *Proceedings of COLING '04*, page 611.

José María Gómez Hidalgo, Guillermo Cajigas Bringas, Enrique Puertas Sánz, and Francisco Carrero García. 2006. Content based sms spam filtering. In *Proceedings of DocEng '06*, pages 107–114.

David I. Holmes. 1998. The evolution of stylometry in humanities scholarship. *Literary and Linguistic Computing*, 13(3):111–117.

Moshe Koppel, Shlomo Argamon, and Anat R. Shimoni. 2003. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of COLING '02*, pages 1–7.

T. C. Mendenhall. 1887. The characteristic curves of composition. *Science*, 9(214):237–246.

A. Q. Morton. 1965. The authorship of greek prose. *Journal of the Royal Statistical Society Series A (General)*, 128(2):169–233.

Frederick Mosteller and David L. Wallace. 1964. *Inference and Disputed Authorship: The Federalist*. Addison-Wesley.

Marina Santini. 2004. A shallow approach to syntactic feature extraction for genre classification. In *Proceedings of CLUK Colloquium '04*.

Dae-Neung Sohn, Joong-Hwi Shin, Jung-Tae Lee, Seung-Wook Lee, and Hae-Chang Rim. 2008. Contents-based korean sms spam filtering using morpheme unit features. In *Proceedings of HCLT '08*, pages 194–199.

E. J. Tweedie, S. Singh, and D. I. Holmes. 1996. Neural network applications in stylometry: The federalist papers. *Computers and the Humanities*, 30:1–10.

G. Udny Yule. 1939. On sentence-length as a statistical characteristic of style in prose, with application to two cases of disputed authorship. *Biometrika*, 30(3-4):363–390.

Learning foci for Question Answering over Topic Maps

Alexander Mikhailian[†], **Tiphaine Dalmas**[‡] and **Rani Pinchuk**[†]
[†]Space Application Services, Leuvensesteenweg 325, B-1932 Zaventem, Belgium
{alexander.mikhailian, rani.pinchuk}@spaceapplications.com
[‡]Aethys
tiphaine.dalmas@aethys.com

Abstract

This paper introduces the concepts of *asking point* and *expected answer type* as variations of the question *focus*. They are of particular importance for QA over semi-structured data, as represented by Topic Maps, OWL or custom XML formats. We describe an approach to the identification of the question *focus* from questions asked to a Question Answering system over Topic Maps by extracting the *asking point* and falling back to the *expected answer type* when necessary. We use known machine learning techniques for *expected answer type* extraction and we implement a novel approach to the *asking point* extraction. We also provide a mathematical model to predict the performance of the system.

1 Introduction

Topic Maps is an ISO standard[1] for knowledge representation and information integration. It provides the ability to store complex meta-data together with the data itself.

This work addresses *domain portable* Question Answering (QA) over Topic Maps. That is, a QA system capable of retrieving answers to a question asked against one particular topic map or topic maps collection at a time. We concentrate on an empirical approach to extract the question *focus*. The extracted focus is then anchored to a topic map construct. This way, we map the type of the answer as provided in the question to the type of the answer as available in the source data.

Our system runs over semi-structured data that encodes ontological information. The classification scheme we propose is based on one dynamic and one static layer, contrasting with previous work that uses static taxonomies (Li and Roth, 2002).

We use the term *asking point* or AP when the type of the answer is explicit, e.g. the word `operas` in the question *What operas did Puccini write?*

We use the term *expected answer type* or EAT when the type of the answer is implicit but can be deduced from the question using formal methods. The question *Who composed Tosca?* implies that the answer is a person. That is, *person* is the *expected answer type*.

We consider that AP takes precedence over the EAT. That is, if the AP (the explicit focus) has been successfully identified in the question, it is considered to be the type of the question, and the EAT (the implicit focus) is left aside.

The claim that the exploitation of AP yields better results in QA over Topic Maps has been tested with 100 questions over the Italian Opera topic map [2]. AP, EAT and the answers of the questions were manually annotated. The answers to the questions were annotated as topic map constructs (i.e. as topics or as occurrences).

An evaluation for QA over Topic Maps has been devised that has shown that choosing APs as foci leads to a much better recall and precision. A detailed description of this test is beyond the scope of this paper.

2 System Architecture

We approach both AP and EAT extraction with the same machine learning technology based on the principle of maximum entropy (Ratnaparkhi, 1998)[3].

[1] ISO/IEC 13250:2003, `http://www.isotopicmaps.org/sam/`

[2] `http://ontopia.net/omnigator/models/topicmap_complete.jsp?tm=opera.ltm`

[3] OpenNLP `http://opennlp.sf.net` was used for tokenization, POS tagging and parsing. Maxent `http://maxent.sf.net` was used as the maximum entropy engine

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 325–328,
Suntec, Singapore, 4 August 2009.

	What	are	Italian	operas	?
Gold	O	O	AP	AP	O

Table 1: Gold standard AP annotation

Class	Word count	%
AskingPoint	1842	9.3%
Other	17997	90.7%

Table 2: Distribution of AP classes (word level)

Class	Count	%
TIME	136	6.5%
NUMERIC	215	10.2%
DEFINITION	281	13.4%
LOCATION	329	15.7%
HUMAN	420	20.0%
OTHER	719	34.2%

Table 3: Distribution of EAT classes (question level)

We annotated a corpus of 2100 questions. 1500 of those questions come from the Li & Roth corpus (Li and Roth, 2002), 500 questions were taken from the TREC-10 questions and 100 questions were asked over the Italian Opera topic map.

2.1 AP extraction

We propose a model for extracting AP that is based on word tagging. As opposed to EAT, AP is constructed on word level not on the question level. Table 1 provides an annotated example of AP.

Our annotation guidelines limit the AP to the noun phrase that is expected to be the type of the answer. As such, it is different from the notion of focus as a noun *likely to be present in the answer* (Ferret et al., 2001) or as *what the question is all about* (Moldovan et al., 1999). For instance, a question such as *Where is the Taj Mahal?* does not yield any AP. Although the main topic is the Taj Mahal, the answer is not expected to be in a parent-child relationship with the subject. Instead, the sought after type is the EAT class LOCATION. This distinction is important for QA over semi-structured data where the data itself is likely to be hierarchically organized.

Asking points were annotated in 1095 (52%) questions out of 2100. The distribution of AP classes in the annotated data is shown in the Table 2.

A study of the inter-annotator agreement between two human annotators has been performed on a set of 100 questions. The Cohen's kappa coefficient (Cohen, 1960) was at 0.781, which is lower than the same measure for the inter-annotator agreement on EAT. This is an expected result, as the AP annotation is naturally perceived as a more complex task. Nevertheless, this allows to qualify the inter-annotator agreement as good.

For each word, a number of features were used by the classifier, including strings and POS-tags on a 4-word window. The WH-word and its complement were also used as features, as well as the parsed subject of the question and the first nominal phrase.

A simple rule-based AP extraction has also been implemented, for comparison. It operates by retrieving the WH-complement from the syntactic parse of the question and stripping the initial articles and numerals, to match the annotation guidelines for AP.

2.2 EAT extraction

EAT was supported by a taxonomy of 6 coarse classes: HUMAN, NUMERIC, TIME, LOCATION, DEFINITION and OTHER. This selection is fairly close to the MUC typology of Named Entities[4] which has been the basis of numerous feature-driven classifiers because of salient formal indices that help identify the correct class.

We purposely limited the number of EAT classes to 6 as AP extraction already provides a fine-grained, dynamic classification from the question to drive the subsequent search in the topic map.

The distribution of EAT classes in the annotated data is shown in the Table 3.

A study of the inter-annotator agreement between two human annotators has been performed on a set of 200 questions. The resulting Cohen's kappa coefficient (Cohen, 1960) of 0.8858 allows to qualify the inter-annotator agreement as very good.

We followed Li & Roth (Li and Roth, 2002) to implement the features for the EAT classifier. They included strings and POS-tags, as well as syntactic parse information (WH-words and their complements, auxiliaries, subjects). Four lists for

for EAT and AP extraction.

[4] http://www.cs.nyu.edu/cs/faculty/grishman/NEtask20.book_1.html

Accuracy	Value	Std dev	Std err
EAT	0.824	0.020	0.006
Lenient AP	0.963	0.020	0.004
Exact AP	0.888	0.052	0.009
Focus (AP+EAT)	0.827	0.020	0.006

Table 4: Accuracy of the classifiers (question level)

words related to locations, people, quantities and time were derived from WordNet and encoded as semantic features.

3 Evaluation Results

The performance of the classifiers was evaluated on our corpus of 2100 questions annotated for AP and EAT. The corpus was split into 80% of training and 20% test data, and data re-sampled 10 times in order to account for variance.

Table 4 lists the figures for the accuracy of the classifiers, that is, the ratio between the correct instances and the overall number of instances. As the AP classifier operates on words while the EAT classifier operates on questions, we had to estimate the accuracy of the AP classifier per question, to allow for comparison. Two simple metrics are possible. A *lenient* metric assumes that the AP extractor performed correctly in the question if there is an overlap between the system output and the annotation on the question level. An *exact* metric assumes that the AP extractor performed correctly if there is an exact match between the system output and the annotation.

In the example *What are Italian Operas?* (Table 1), assuming the system only tagged *operas* as AP, lenient accuracy will be 1, exact accuracy will be 0, precision for the AskingPoint class will be 1 and its recall will be 0.5.

Table 5 shows EAT results by class. Tables 6 and 7 show AP results by class for the machine learning and the rule-based classifier.

As shown in Figure 1, when AP classification is available it is used. During the evaluation, AP was found in 49.4% of questions.

A mathematical model has been devised to predict the accuracy of the focus extractor on an annotated corpus.

It is expected that the focus accuracy, that is, the accuracy of the focus extraction system, is dependent on the performance of the AP and the EAT classifiers. Given N the total number of questions,

Class	Precision	Recall	F-Score
DEFINITION	0.887	0.800	0.841
LOCATION	0.834	0.812	0.821
HUMAN	0.902	0.753	0.820
TIME	0.880	0.802	0.838
NUMERIC	0.943	0.782	0.854
OTHER	0.746	0.893	0.812

Table 5: EAT performance by class (question level)

Class	Precision	Recall	F-Score
AskingPoint	0.854	0.734	0.789
Other	0.973	0.987	0.980

Table 6: AP performance by class (word level)

Class	Precision	Recall	F-Score
AskingPoint	0.608	0.479	0.536
Other	0.948	0.968	0.958

Table 7: Rule-based AP performance by class (word level)

we define the branching factor, that is, the percentage of questions for which AP is provided by the system, as follows:

$$Y = \frac{(TP_{AP} + FP_{AP})}{N}$$

Figure 1 shows that the sum AP true positives and EAT correct classifications represents the overall number of questions that were classified correctly. This accuracy can be further developed to present the dependencies as follows:

$$A_{FOCUS} = P_{AP}Y + A_{EAT}(1 - Y)$$

That is, the overall accuracy is dependent on the precision of the AskingPoint class of the AP classifier, the accuracy of EAT and the branching factor. The branching factor itself can be predicted using the performance of the AP classifier and the ratio between the number of questions annotated with AP and the total number of questions.

$$Y = \frac{(\frac{TP_{AP}+FN_{AP}}{N})R_{AP}}{P_{AP}}$$

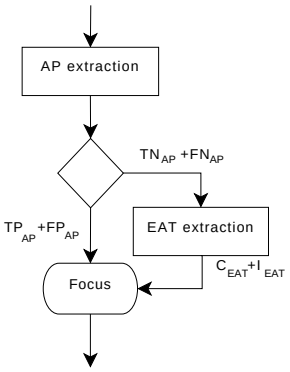


Figure 1: Focus extraction flow diagram

4 Related work

(Atzeni et al., 2004; Paggio et al., 2004) describe MOSES, a multilingual QA system delivering answers from Topic Maps. MOSES extracts a focus constraint (defined after (Rooth, 1992)) as part of the question analysis, which is evaluated to an accuracy of 76% for the 85 Danish questions and 70% for the 83 Italian questions. The focus is an ontological type dependent from the topic map, and its extraction is based on hand-crafted rules. In our case, focus extraction – though defined with topic map retrieval in mind – stays clear of ontological dependencies so that the same question analysis module can be applied to any topic map.

In open domain QA, machine learning approaches have proved successful since Li & Roth (Li and Roth, 2006). Despite using similar features, the F-Score (0.824) for our EAT classes is slightly lower than reported by Li & Roth (Li and Roth, 2006) for coarse classes. We may speculate that the difference is primarily due to our limited training set size (1,680 questions versus 21,500 questions for Li & Roth). On the other hand, we are not aware of any work attempting to extract AP on word level using machine learning in order to provide dynamic classes to a question classification module.

5 Future work and conclusion

We presented a question classification system based on our definition of *focus* geared towards QA over semi-structured data where there is a parent-child relationship between answers and their types. The specificity of the focus degrades gracefully in the approach described above. That is, we attempt the extraction of the AP when possible and fall back on the EAT extraction otherwise.

We identify the focus dynamically, instead of relying on a static taxonomy of question types, and we do so using machine learning techniques throughout the application stack.

A mathematical model has been devised to predict the performance of the focus extractor.

We are currently working on the exploitation of the results provided by the focus extractor in the subsequent modules of the QA over Topic Maps, namely anchoring, navigation in the topic map, graph algorithms and reasoning.

Acknowledgements

This work has been partly funded by the Flemish government (through IWT) as part of the ITEA2 project LINDO (ITEA2-06011).

References

P. Atzeni, R. Basili, D. H. Hansen, P. Missier, P. Paggio, M. T. Pazienza, and F. M. Zanzotto. 2004. Ontology-Based Question Answering in a Federation of University Sites: The MOSES Case Study. In *NLDB*, pages 413–420.

J. Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20, No.1:37–46.

O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, I. Robba, and A. Vilnat. 2001. Finding an Answer Based on the Recognition of the Question Focus. In *10th Text Retrieval Conference*.

X. Li and D. Roth. 2002. Learning Question Classifiers. In *19th International Conference on Computational Linguistics (COLING)*, pages 556–562.

X. Li and D. Roth. 2006. Learning Question Classifiers: The Role of Semantic Information. *Journal of Natural Language Engineering*, 12(3):229–250.

D. Moldovan, S. Harabagiu, M. Pasca, R. Mihalcea, R. Goodrum, R. Girju, and V. Rus. 1999. LASSO: A Tool for Surfing the Answer Net. In *8th Text Retrieval Conference*.

P. Paggio, D. H. Hansen, R. Basili, M. T. Pazienza, and F. M. Zanzotto. 2004. Ontology-based question analysis in a multilingual environment: the MOSES case study. In *OntoLex (LREC)*.

A. Ratnaparkhi. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

M. Rooth. 1992. A Theory of Focus Interpretation. *Natural Language Semantics*, 1(1):75–116.

Do Automatic Annotation Techniques Have Any Impact on Supervised Complex Question Answering?

Yllias Chali
University of Lethbridge
Lethbridge, AB, Canada
chali@cs.uleth.ca

Sadid A. Hasan
University of Lethbridge
Lethbridge, AB, Canada
hasan@cs.uleth.ca

Shafiq R. Joty
University of British Columbia
Vancouver, BC, Canada
rjoty@cs.ubc.ca

Abstract

In this paper, we analyze the impact of different automatic annotation methods on the performance of supervised approaches to the complex question answering problem (defined in the DUC-2007 main task). Huge amount of annotated or labeled data is a prerequisite for supervised training. The task of labeling can be accomplished either by humans or by computer programs. When humans are employed, the whole process becomes time consuming and expensive. So, in order to produce a large set of labeled data we prefer the automatic annotation strategy. We apply five different automatic annotation techniques to produce labeled data using ROUGE similarity measure, Basic Element (BE) overlap, syntactic similarity measure, semantic similarity measure, and Extended String Subsequence Kernel (ESSK). The representative supervised methods we use are Support Vector Machines (SVM), Conditional Random Fields (CRF), Hidden Markov Models (HMM), and Maximum Entropy (MaxEnt). Evaluation results are presented to show the impact.

1 Introduction

In this paper, we consider the complex question answering problem defined in the DUC-2007 main task[1]. We focus on an extractive approach of summarization to answer complex questions where a subset of the sentences in the original documents are chosen. For supervised learning methods, huge amount of annotated or labeled data sets are obviously required as a precondition. The decision as to whether a sentence is important enough to be annotated can be taken either by humans or by computer programs. When humans are employed in the process, producing such a large labeled corpora becomes time consuming and expensive. There comes the necessity of using automatic methods to align sentences with the intention to build extracts from abstracts. In this paper, we use ROUGE similarity measure, Basic Element (BE) overlap, syntactic similarity measure, semantic similarity measure, and Extended String Subsequence Kernel (ESSK) to automatically label the corpora of sentences (DUC-2006 data) into extract summary or non-summary categories in correspondence with the document abstracts. We feed these 5 types of labeled data into the learners of each of the supervised approaches: SVM, CRF, HMM, and MaxEnt. Then we extensively investigate the performance of the classifiers to label unseen sentences (from 25 topics of DUC-2007 data set) as summary or non-summary sentence. The experimental results clearly show the impact of different automatic annotation methods on the performance of the candidate supervised techniques.

[1] http://www-nlpir.nist.gov/projects/duc/duc2007/

2 Automatic Annotation Schemes

Using ROUGE Similarity Measures ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is an automatic tool to determine the quality of a summary using a collection of measures ROUGE-N (N=1,2,3,4), ROUGE-L, ROUGE-W and ROUGE-S which count the number of overlapping units such as n-gram, word-sequences, and word-pairs between the extract and the abstract summaries (Lin, 2004). We assume each individual document sentence as the extract summary and calculate its ROUGE similarity scores with the corresponding abstract summaries. Thus an average ROUGE score is assigned to each sentence in the document. We choose the top N sentences based on ROUGE scores to have the label

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 329–332,
Suntec, Singapore, 4 August 2009.

$+1$ (summary sentences) and the rest to have the label -1 (non-summary sentences).

Basic Element (BE) Overlap Measure We extract BEs, the "head-modifier-relation" triples for the sentences in the document collection using BE package 1.0 distributed by ISI [2]. The ranked list of BEs sorted according to their Likelihood Ratio (LR) scores contains important BEs at the top which may or may not be relevant to the abstract summary sentences. We filter those BEs by checking possible matches with an *abstract sentence word* or a *related word*. For each abstract sentence, we assign a score to every document sentence as the sum of its filtered BE scores divided by the number of BEs in the sentence. Thus, every abstract sentence contributes to the BE score of each document sentence and we select the top N sentences based on average BE scores to have the label $+1$ and the rest to have the label -1.

Syntactic Similarity Measure In order to calculate the syntactic similarity between the abstract sentence and the document sentence, we first parse the corresponding sentences into syntactic trees using Charniak parser [3] (Charniak, 1999) and then we calculate the similarity between the two trees using the *tree kernel* (Collins and Duffy, 2001). We convert each parenthesis representation generated by Charniak parser to its corresponding tree and give the trees as input to the tree kernel functions for measuring the syntactic similarity. The tree kernel of two syntactic trees T_1 and T_2 is actually the inner product of the two m-dimensional vectors, $v(T_1)$ and $v(T_2)$:

$$TK(T_1, T_2) = v(T_1).v(T_2)$$

The *TK* (tree kernel) function gives the similarity score between the abstract sentence and the document sentence based on the syntactic structure. Each abstract sentence contributes a score to the document sentences and the top N sentences are selected to be annotated as $+1$ and the rest as -1 based on the average of similarity scores.

Semantic Similarity Measure Shallow semantic representations, bearing a more compact information, can prevent the sparseness of deep structural approaches and the weakness of BOW models (Moschitti et al., 2007). To experiment with semantic structures, we parse the corresponding sentences semantically using a Semantic Role Labeling (SRL) system like ASSERT[4]. ASSERT is an automatic statistical semantic role tagger, that can annotate naturally occuring text with semantic arguments. We represent the annotated sentences using tree structures called semantic trees (ST). Thus, by calculating the similarity between STs, each document sentence gets a semantic similarity score corresponding to each abstract sentence and then the top N sentences are selected to be labeled as $+1$ and the rest as -1 on the basis of average similarity scores.

Extended String Subsequence Kernel (ESSK) Formally, ESSK is defined as follows (Hirao et al., 2004):

$$K_{essk}(T, U) = \sum_{m=1}^{d} \sum_{t_i \in T} \sum_{u_j \in U} K_m(t_i, u_j)$$

$$K_m(t_i, u_j) = \begin{cases} val(t_i, u_j) & \text{if } m = 1 \\ K'_{m-1}(t_i, u_j) \cdot val(t_i, u_j) & \end{cases}$$

Here, $K'_m(t_i, u_j)$ is defined below. t_i and u_j are the nodes of T and U, respectively. Each node includes a word and its disambiguated sense. The function $val(t, u)$ returns the number of attributes common to the given nodes t and u.

$$K'_m(t_i, u_j) = \begin{cases} 0 & \text{if } j = 1 \\ \lambda K'_m(t_i, u_{j-1}) + K''_m(t_i, u_{j-1}) & \end{cases}$$

Here λ is the decay parameter for the number of skipped words. We choose $\lambda = 0.5$ for this research. $K''_m(t_i, u_j)$ is defined as:

$$K''_m(t_i, u_j) = \begin{cases} 0 & \text{if } i = 1 \\ \lambda K''_m(t_{i-1}, u_j) + K_m(t_{i-1}, u_j) & \end{cases}$$

Finally, the similarity measure is defined after normalization as below:

$$sim_{essk}(T, U) = \frac{K_{essk}(T, U)}{\sqrt{K_{essk}(T, T) K_{essk}(U, U)}}$$

Indeed, this is the similarity score we assign to each document sentence for each abstract sentence and in the end, top N sentences are selected to be annotated as $+1$ and the rest as -1 based on average similarity scores.

3 Experiments

Task Description The problem definition at DUC-2007 was: *"Given a complex question (topic description) and a collection of relevant documents, the task is to synthesize a fluent, well-organized 250-word summary of the documents*

[2] BE website:http://www.isi.edu/ cyl/BE

[3] available at ftp://ftp.cs.brown.edu/pub/nlparser/

[4] available at http://cemantix.org/assert

that answers the question(s) in the topic". We consider this task and use the five automatic annotation methods to label each sentence of the 50 document sets of DUC-2006 to produce five different versions of training data for feeding the SVM, HMM, CRF and MaxEnt learners. We choose the top 30% sentences (based on the scores assigned by an annotation scheme) of a document set to have the label $+1$ and the rest to have -1. Unlabeled sentences of 25 document sets of DUC-2007 data are used for the testing purpose.

Feature Space We represent each of the document-sentences as a vector of feature-values. We extract several query-related features and some other important features from each sentence. We use the features: n-gram overlap, Longest Common Subsequence (LCS), Weighted LCS (WLCS), skip-bigram, exact word overlap, synonym overlap, hypernym/hyponym overlap, gloss overlap, Basic Element (BE) overlap, syntactic tree similarity measure, position of sentences, length of sentences, Named Entity (NE), cue word match, and title match (Edmundson, 1969).

Supervised Systems For SVM we use second order polynomial kernel for the ROUGE and ESSK labeled training. For the BE, syntactic, and semantic labeled training third order polynomial kernel is used. The use of kernel is based on the accuracy we achieved during training. We apply 3-fold cross validation with randomized local-grid search for estimating the value of the trade-off parameter C. We try the value of C in 2^i following heuristics, where $i \in \{-5, -4, \cdots, 4, 5\}$ and set C as the best performed value 0.125 for second order polynomial kernel and default value is used for third order kernel. We use SVM^{light} [5] package for training and testing in this research. In case of HMM, we apply the Maximum Likelihood Estimation (MLE) technique by frequency counts with add-one smoothing to estimate the three HMM parameters: initial state probabilities, transition probabilities and emission probabilities. We use Dr. Dekang Lin's HMM package[6] to generate the most probable label sequence given the model parameters and the observation sequence (unlabeled DUC-2007 test data). We use MALLET-0.4 NLP toolkit[7] to implement the CRF. We formulate our problem in terms of MALLET's SimpleTagger class which is a command line interface to the MALLET CRF class. We modify the SimpleTagger class in order to include the provision for producing corresponding posterior probabilities of the predicted labels which are used later for ranking sentences. We build the MaxEnt system using Dr. Dekang Lin's MaxEnt package[8]. To define the exponential prior of the λ values in MaxEnt models, an extra parameter α is used in the package during training. We keep the value of α as default.

Sentence Selection The proportion of important sentences in the training data will differ from the one in the test data. A simple strategy is to rank the sentences in a document, then select the top N sentences. In SVM systems, we use the normalized distance from the hyperplane to each sample to rank the sentences. Then, we choose N sentences until the summary length (250 words for DUC-2007) is reached. For HMM systems, we use Maximal Marginal Relevance (MMR) based method to rank the sentences (Carbonell et al., 1997). In CRF systems, we generate posterior probabilities corresponding to each predicted label in the label sequence to measure the confidence of each sentence for summary inclusion. Similarly for MaxEnt, the corresponding probability values of the predicted labels are used to rank the sentences.

Evaluation Results The multiple "reference summaries" given by DUC-2007 are used in the evaluation of our summary content. We evaluate the system generated summaries using the automatic evaluation toolkit ROUGE (Lin, 2004). We report the three widely adopted important ROUGE metrics in the results: ROUGE-1 (unigram), ROUGE-2 (bigram) and ROUGE-SU (skip bi-gram). Figure 1 shows the ROUGE F-measures for SVM, HMM, CRF and MaxEnt systems. The X-axis containing **ROUGE**, **BE**, **Synt** (Syntactic), **Sem** (Semantic), and **ESSK** stands for the annotation scheme used. The Y-axis shows the ROUGE-1 scores at the top, ROUGE-2 scores at the bottom and ROUGE-SU scores in the middle. The supervised systems are distinguished by the line style used in the figure.

From the figure, we can see that the *ESSK* labeled SVM system is having the poorest ROUGE-1 score whereas the *Sem* labeled system performs

[5] http://svmlight.joachims.org/

[6] http://www.cs.ualberta.ca/ lindek/hmm.htm

[7] http://mallet.cs.umass.edu/

[8] http://www.cs.ualberta.ca/lindek/downloads.htm

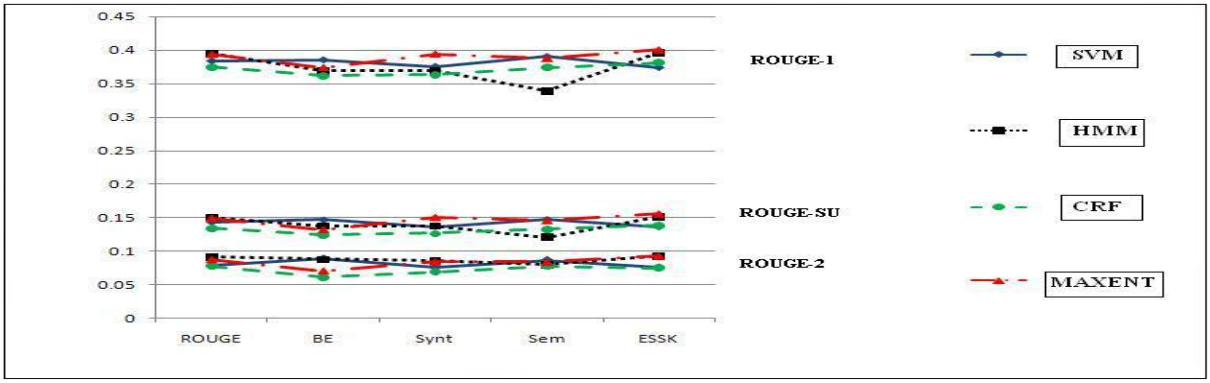


Figure 1: ROUGE F-scores for different supervised systems

best. The other annotation methods' impact is almost similar here in terms of ROUGE-1. Analyzing ROUGE-2 scores, we find that the *BE* performs the best for SVM, on the other hand, *Sem* achieves top ROUGE-SU score. As for the two measures *Sem* annotation is performing the best, we can typically conclude that *Sem* annotation is the most suitable method for the SVM system. *ESSK* works as the best for HMM and *Sem* labeling performs the worst for all ROUGE scores. *Synt* and *BE* labeled HMMs perform almost similar whereas *ROUGE* labeled system is pretty close to that of *ESSK*. Again, we see that the CRF performs best with the *ESSK* annotated data in terms of ROUGE -1 and ROUGE-SU scores and *Sem* has the highest ROUGE-2 score. But *BE* and *Synt* labeling work bad for CRF whereas the *ROUGE* labeling performs decently. So, we can typically conclude that *ESSK* annotation is the best method for the CRF system. Analyzing further, we find that *ESSK* works best for MaxEnt and *BE* labeling is the worst for all ROUGE scores. We can also see that *ROUGE*, *Synt* and *Sem* labeled MaxEnt systems perform almost similar. So, from this discussion we can come to a conclusion that SVM system performs best if the training data uses semantic annotation scheme and *ESSK* works best for HMM, CRF and MaxEnt systems.

4 Conclusion and Future Work

In the work reported in this paper, we have performed an extensive experimental evaluation to show the impact of five automatic annotation methods on the performance of different supervised machine learning techniques in confronting the complex question answering problem. Experimental results show that *Sem* annotation is the best for SVM whereas *ESSK* works well for HMM, CRF and MaxEnt systems. In the near future, we plan to work on finding more sophisticated approaches to effective automatic labeling so that we can experiment with different supervised methods.

References

Jaime Carbonell, Yibing Geng, and Jade Goldstein. 1997. Automated query-relevant summarization and diversity-based reranking. In *IJCAI-97 Workshop on AI in Digital Libraries*, pages 12–19, Japan.

Eugene Charniak. 1999. A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department.

Michael Collins and Nigel Duffy. 2001. Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems*, pages 625–632, Vancouver, Canada.

Harold P. Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285.

Tsutomu Hirao, Jun Suzuki, Hideki Isozaki, and Eisaku Maeda. 2004. Dependency-based sentence alignment for multiple document summarization. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 446–452.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of Association for Computational Linguistics*, pages 74–81, Barcelona, Spain.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classificaion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic. ACL.

Where's the Verb?
Correcting Machine Translation During Question Answering

Wei-Yun Ma, Kathleen McKeown
Department of Computer Science
Columbia University
New York, NY 10027, USA
{ma,kathy}@cs.columbia.edu

Abstract

When a multi-lingual question-answering (QA) system provides an answer that has been incorrectly translated, it is very likely to be regarded as irrelevant. In this paper, we propose a novel method for correcting a deletion error that affects overall understanding of the sentence. Our post-editing technique uses information available at query time: examples drawn from related documents determined to be relevant to the query. Our results show that 4%-7% of MT sentences are missing the main verb and on average, 79% of the modified sentences are judged to be more comprehensible. The QA performance also benefits from the improved MT: 7% of irrelevant response sentences become relevant.

1. Introduction

We are developing a multi-lingual question-answering (QA) system that must provide relevant English answers for a given query, drawing pieces of the answer from translated foreign source. Relevance and translation quality are usually inseparable: an incorrectly translated sentence in the answer is very likely to be regarded as irrelevant even when the corresponding source language sentence is actually relevant. We use a phrase-based statistical machine translation system for the MT component and thus, for us, MT serves as a black box that produces the translated documents in our corpus; we cannot change the MT system itself. As MT is used in more and more multi-lingual applications, this situation will become quite common.

We propose a novel method which uses redundant information available at question-answering time to correct errors. We present a post-editing mechanism to both detect and correct errors in translated documents determined to be relevant for the response. In this paper, we focus on cases where the main verb of a Chinese sentence has not been translated. The main verb usually plays a crucial role in conveying the meaning of a sentence. In cases where only the main verb is missing, an MT score relying on edit distance (e.g., TER or Bleu) may be high, but the sentence may nonetheless be incomprehensible.

Handling this problem at query time rather than during SMT gives us valuable information which was not available during SMT, namely, a set of related sentences and their translations which may contain the missing verb. By using translation examples of verb phrases and alignment information in the related documents, we are able to find an appropriate English verb and embed it in the right position as the main verb in order to improve MT quality.

A missing main verb can result in an incomprehensible sentence as seen here where the Chinese verb "被捕" was not translated at all.

MT: On December 13 Saddam .
REF : On December 13 Saddam **was arrested**.
Chinese: 12月13日萨达姆**被捕**。

In other cases, a deleted main verb can result in miscommunication; below the Chinese verb "减退" should have been translated as "reduced". An English native speaker could easily misunderstand the meaning to be "People love classical music every year." which happens to be the opposite of the original intended meaning.

MT: People of classical music loving every year.
REF : People's love for classical music **reduced** every year.
Chinese: 民众对古典音乐的热爱逐年**减退**。

2. Related Work

Post-editing has been used in full MT systems for tasks such as article selection (a, an, the) for

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 333–336,
Suntec, Singapore, 4 August 2009.

English noun phrases (Knight and Chander 1994). Simard et al in 2007 even developed a statistical phrase based MT system in a post-editing task, which takes the output of a rule-based MT system and produces post-edited target-language text. Zwarts et al. (2008) target selecting the best of a set of outputs from different MT systems through their classification-based approach. Others have also proposed using the question-answering context to detect errors in MT, showing how to correct names (Parton et. al 2008, Ji et. al 2008).

3. System Overview

The architecture of our QA system is shown in Figure 1. Our MT post-editing system (the bold block in Figure 1) runs after document retrieval has retrieved all potentially relevant documents and before the response generator selects sentences for the answer. It modifies any MT documents retrieved by the embedded information retrieval system that are missing a main verb. All MT results are provided by a phrase-based SMT system.

Post-editing includes three steps: detect a clause with a missing main verb, determine which Chinese verb should have been translated, and find an example sentence in the related documents with an appropriate sentence which can be used to modify the sentence in question. To detect clauses, we first tag the corpus using a Conditional Random Fields (CRF) POS tagger and then use manually designed regular expressions to identify main clauses of the sentence, subordinate clauses (i.e., clauses which are arguments to a verb) and conjunct clauses in a sentence with conjunction. We do not handle adjunct clauses. Hereafter, we simply refer to all of these as "clause". If a clause does not have any POS tag that can serve as a main verb (VB, VBD, VBP, VBZ), it is marked as missing a main verb.

MT alignment information is used to further ensure that these marked clauses are really missing main verbs. We segment and tag the Chinese source sentence using the Stanford Chinese segmenter and the CRF Chinese POS tagger developed by Purdue University. If we find a verb phrase in the Chinese source sentence that was not aligned with any English words in the SMT alignment tables, then we label it as a verb translation gap (VTG) and confirm that the marking was correct.

In the following sections, we describe how we determine which Chinese verb should have been translated and how that occurs.

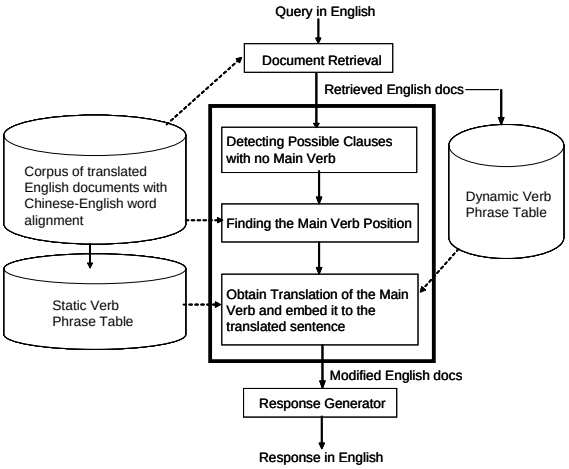


Figure 1. The System Pipeline

4. Finding the Main Verb Position

Chinese ordering differs from English mainly in clause ordering (Wang et al., 2007) and within the noun phrase. But within a clause centered by a verb, Chinese mostly uses a SVO or SV structure, like English (Yamada and Knight 2001), and we can assume the local alignment centered by a verb between Chinese and English is a linear mapping relation. Under this assumption, the translation of "被捕" in the above example should be placed in the position between "Saddam" and ".". Thus, once we find a VTG, its translation can be inserted into the corresponding position of the target sentence using the alignment.

This assumes, however, that there is only one VTG found within a clause. In practice, more than one VTG may be found in a clause. If we choose one of them, we risk making the wrong choice. Instead, we insert the translations of both VTGs simultaneously. This strategy could result in more than one main verb in a clause, but it is more helpful than having no verb at all.

5. Obtaining a VTG Translation

We translate VTGs by using verb redundancy in related documents: if the VTG was translated in other places in related documents, the existing translations can be reused. Related documents are likely to use a good translation for a specific VTG as it is used in a similar context. A verb's aspect and tense can be directly determined by referencing the corresponding MT examples and their contexts. If, unfortunately, a given VTG

did not have any other translation record, then the VTG will not be processed.

To do this, our system first builds verb phrase tables from relevant documents and then uses the tables to translate the VTG. We use two verb phrase tables: one is built from a collection of MT documents before any query and is called the "Static Verb Phrase Table", and the other one is dynamically built from the retrieved relevant MT documents for each query and is called the "Dynamic Verb Phrase Table".

The construction procedure is the same for both. Given a set of related MT documents and their MT alignments, we collect all Chinese verb phrases and their translations along with their frequencies and contexts.

One key issue is to decide appropriate contextual features of a verb. A number of researchers (Cabezas and Resnik 2005, Carpuat and Wu 2007) provide abundant evidence that rich context features are useful in MT tasks. Carpuat and Wu (2007) tried to integrate a Phrase Sense Disambiguation (PSD) model into their Chinese-English SMT system and they found that the POS tag preceding a given phrase, the POS tag following the phrase and bag-of-words are the three most useful features. Following their approach, we use the word preceding and the word following a verb as the context features.

The Static and Dynamic Verb Phrase Tables provide us with MT examples to translate a VTG. The system first references the Dynamic Verb Phrase Table as it is more likely to yield a good translation. If the record is not found, the Static one is referenced. If it is not found in either, the given VTG will not be processed. No matter which table is referenced, the following Naive Bayes equation is applied to obtain the translation of a given VTG.

$$t' = \arg\max_{t_k} P(t_k \mid pw, fw)$$
$$= \arg\max_{t_k} (\log P(t_k) + \log P(pw \mid t_k) + \log P(fw \mid t_k))$$

pw, fw and t_k respectively represent the preceding source word, the following source word and a translation candidate of a VTG.

6. Experiments

Our test data is drawn from Chinese-English MT results generated by Aachen's 2007 RWTH system (Mauser et al., 2007), a phrase-based SMT system with 38.5% BLEU score on IWSLT 2007 evaluation data.

Newswires and blog articles are retrieved for five queries which served as our experimental test bed. The queries are open-ended and on average, answers were 30 sentences in length.

Q1: Who/What is involved in Saddam Hussein's trial
Q2: Produce a biography of Jacques Rene Chirac
Q3: Describe arrests of person from Salafist Group for Preaching and Combat
Q4: Provide information on Chen Sui Bian
Q5: What connections are there between World Cup games and stock markets?

We used MT documents retrieved by IR for each query to build the Dynamic Verb Phrase Table. We tested the system on 18,886 MT sentences from the retrieved MT documents for all of the five queries. Among these MT sentences, 1,142 sentences were detected and modified (6 % of all retrieved MT sentences).

6.1 Evaluation Methodology

For evaluation, we used human judgments of the modified and original MT. We did not have reference translations for the data used by our question-answering system and thus, could not use metrics such as TER or Bleu. Moreover, at best, TER or Bleu score would increase by a small amount and that is only if we select the same main verb in the same position as the reference. Critically, we also know that a missing main verb can cause major problems with comprehension. Thus, readers could better determine if the modified sentence better captured the meaning of the source sentence. We also evaluated relevance of a sentence to a query before and after modification.

We recruited 13 Chinese native speakers who are also proficient in English to judge MT quality. Native English speakers cannot tell which translation is better since they do not understand the meaning of the original Chinese. To judge relevance to the query, we used native English speakers.

Each modified sentence was evaluated by three people. They were shown the Chinese sentence and two translations, the original MT and the modified one. Evaluators did not know which MT sentence was modified. They were asked to decide which sentence is a better translation, after reading the Chinese sentence. An evaluator also had the option of answering "no difference".

6.2 Results and Discussion

We used majority voting (two out of three) to decide the final evaluation of a sentence judged by three people. On average, 900 (79%) of the

1142 modified sentences, which comprise 5% of all 18,886 retrieved MT sentences, are better than the original sentences based on majority voting. And for 629 (70%) of these 900 better modified sentences all three evaluators agreed that the modified sentence is better.

Furthermore, we found that for every individual query, the evaluators preferred more of the modified sentences than the original MT. And among these improved sentences, 81% sentences reference the Dynamic Verb Phrase Table, while only 19% sentences had to draw from the Static Verb Phrase Table, thus demonstrating that the question answering context is quite helpful in improving MT.

We also evaluated the impact of post-editing on the 234 sentences returned by our response generator. In our QA task, response sentences were judged as "Relevant(R)", "Partially Relevant(PR)", "Irrelevant(I)" and "Too little information to judge(T)" sentences. With our post-editing technique, 7% of 141 I/T responses become R/PR responses and none of the R/PR responses become I/T responses. This means that R/PR response percentage has an increase of 4%, thus demonstrating that our correction of MT truly improves QA performance. An example of a change from T to PR is:

Question: What connections are there between World Cup games and stock markets?
Original QA answer: But if winning the ball, not necessarily in the stock market.
Modified QA answer: But if winning the ball, not necessarily in the stock market ***increased***.

6.3 Analysis of Different MT Systems

In order to examine how often missing verbs occur in different recent MT systems, in addition to using Aachen's up-to-date system – "RWTH-PBT"of 2008, we also ran the detection process for another state-of-the-art MT system – "SRI-HPBT" (Hierarchical Phrase-Based System) of 2008 provided by SRI, which uses a grammar on the target side as well as reordering, and focuses on improving grammaticality of the target language. Based on a government 2008 MT evaluation, the systems achieve 30.3% and 30.9% BLEU scores respectively. We used the same test set, which includes 94 written articles (953 sentences).

Overall, 7% of sentences translated by RWTH-PBT are detected with missing verbs while 4% of sentences translated by SRI-HPBT are detected with missing verb. This shows that while MT systems improve every year, missing verbs remain a problem.

7 Conclusions

In this paper, we have presented a technique for detecting and correcting deletion errors in translated Chinese answers as part of a multi-lingual QA system. Our approach uses a regular grammar and alignment information to detect missing verbs and draws from examples in documents determined to be relevant to the query to insert a new verb translation. Our evaluation demonstrates that MT quality and QA performance are both improved. In the future, we plan to extend our approach to tackle other MT error types by using information available at query time.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency under Contract No. HR0011-06-C-0023

References

Clara Cabezas and Philip Resnik. 2005. *Using WSD Techniques for Lexical Selection in Statistical Machine*, Translation Technical report CS-TR-4736

Marine Carpuat and Dekai Wu. 2007. *Context-Dependent Phrasal Translation Lexicons for Statistical Machine Translation*, Machine Translation Summit XI, Copenhagen

Heng Ji, Ralph Grishman and Wen Wang. 2008. *Phonetic Name Matching For Cross-lingual Spoken Sentence Retrieval*, IEEE-ACL SLT08. Goa, India

K. Knight and I. Chander. 1994. *Automated Postediting of Documents*, AAAI

Kristen Parton, Kathleen R. McKeown, James Allan, and Enrique Henestroza. 2008. *Simultaneous multilingual search for translingual information retrieval*, ACM 17th CIKM

Arne Mauser, David Vilar, Gregor Leusch, Yuqi Zhang, and Hermann Ney. 2007. *The RWTH Machine Translation System for IWSLT 2007*, IWSLT

Michel Simard, Cyril Goutte and Pierre Isabelle. 2007. *Statistical Phrase-based Post-Editing*, NAACL-HLT

Chao Wang, Michael Collins, and Philipp Koehn. 2007. *Chinese Syntactic Reordering for Statistical Machine Translation*, EMNLP-CoNLL.

Kenji Yamada , Kevin Knight. 2001. *A syntax-based statistical translation model*, ACL

S. Zwarts and M. Dras. 2008. *Choosing the Right Translation: A Syntactically Informed Approach*, COLING

A Note on the Implementation of Hierarchical Dirichlet Processes

Phil Blunsom[*]
pblunsom@inf.ed.ac.uk

Trevor Cohn[*]
tcohn@inf.ed.ac.uk

Sharon Goldwater[*]
sgwater@inf.ed.ac.uk

Mark Johnson[†]
mark_johnson@brown.edu

[*]Department of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK

[†]Department of Cognitive and Linguistic Sciences
Brown University
Providence, RI, USA

Abstract

The implementation of collapsed Gibbs samplers for non-parametric Bayesian models is non-trivial, requiring considerable book-keeping. Goldwater et al. (2006a) presented an approximation which significantly reduces the storage and computation overhead, but we show here that their formulation was incorrect and, even after correction, is grossly inaccurate. We present an alternative formulation which is exact and can be computed easily. However this approach does not work for hierarchical models, for which case we present an efficient data structure which has a better space complexity than the naive approach.

1 Introduction

Unsupervised learning of natural language is one of the most challenging areas in NLP. Recently, methods from nonparametric Bayesian statistics have been gaining popularity as a way to approach unsupervised learning for a variety of tasks, including language modeling, word and morpheme segmentation, parsing, and machine translation (Teh et al., 2006; Goldwater et al., 2006a; Goldwater et al., 2006b; Liang et al., 2007; Finkel et al., 2007; DeNero et al., 2008). These models are often based on the Dirichlet process (DP) (Ferguson, 1973) or hierarchical Dirichlet process (HDP) (Teh et al., 2006), with Gibbs sampling as a method of inference. Exact implementation of such sampling methods requires considerable bookkeeping of various counts, which motivated Goldwater et al. (2006a) (henceforth, GGJ06) to develop an approximation using expected counts. However, we show here that their approximation is flawed in two respects: 1) It omits an important factor in the expectation, and 2) Even after correction, the approximation is poor for hierarchical models, which are commonly used for NLP applications. We derive an improved $\mathcal{O}(1)$ formula that gives exact values for the expected counts in non-hierarchical models. For hierarchical models, where our formula is not exact, we present an efficient method for sampling from the HDP (and related models, such as the hierarchical Pitman-Yor process) that considerably decreases the memory footprint of such models as compared to the naive implementation.

As we have noted, the issues described in this paper apply to models for various kinds of NLP tasks; for concreteness, we will focus on n-gram language modeling for the remainder of the paper, closely following the presentation in GGJ06.

2 The Chinese Restaurant Process

GGJ06 present two nonparametric Bayesian language models: a DP unigram model and an HDP bigram model. Under the DP model, words in a corpus $\mathbf{w} = w_1 \dots w_n$ are generated as follows:

$$\begin{aligned} G|\alpha_0, P_0 &\sim \text{DP}(\alpha_0, P_0) \\ w_i|G &\sim G \end{aligned}$$

where G is a distribution over an infinite set of possible words, P_0 (the *base distribution* of the DP) determines the probability that an item will be in the support of G, and α_0 (the *concentration parameter*) determines the variance of G.

One way of understanding the predictions that the DP model makes is through the Chinese restaurant process (CRP) (Aldous, 1985). In the CRP, customers (word tokens w_i) enter a restaurant with an infinite number of tables and choose a seat. The table chosen by the ith customer, z_i, follows the distribution:

$$P(z_i = k|\mathbf{z}_{-i}) = \begin{cases} \frac{n_k^{\mathbf{z}_{-i}}}{i-1+\alpha_0}, 0 \le k < K(\mathbf{z}_{-i}) \\ \frac{\alpha_0}{i-1+\alpha_0}, k = K(\mathbf{z}_{-i}) \end{cases}$$

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 337–340,
Suntec, Singapore, 4 August 2009.

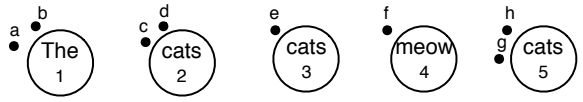


Figure 1. A seating assignment describing the state of a unigram CRP. Letters and numbers uniquely identify customers and tables. Note that multiple tables may share a label.

where $\mathbf{z}_{-i} = z_1 \dots z_{i-1}$ are the table assignments of the previous customers, $n_k^{\mathbf{z}_{-i}}$ is the number of customers at table k in $\mathbf{z}_{-i}$, and $K(\mathbf{z}_{-i})$ is the total number of occupied tables. If we further assume that table k is labeled with a word type ℓ_k drawn from P_0, then the assignment of tokens to tables defines a distribution over words, with $w_i = \ell_{z_i}$. See Figure 1 for an example seating arrangement.

Using this model, the predictive probability of w_i, conditioned on the previous words, can be found by summing over possible seating assignments for w_i, and is given by

$$P(w_i = w|\mathbf{w}_{-i}) = \frac{n_w^{\mathbf{w}_{-i}} + \alpha_0 P_0}{i - 1 + \alpha_0} \qquad (1)$$

This prediction turns out to be exactly that of the DP model after integrating out the distribution G. Note that as long as the base distribution P_0 is fixed, predictions do not depend on the seating arrangement $\mathbf{z}_{-i}$, only on the count of word w in the previously observed words ($n_w^{\mathbf{w}_{-i}}$). However, in many situations, we may wish to estimate the base distribution itself, creating a *hierarchical* model. Since the base distribution generates table labels, estimates of this distribution are based on the counts of those labels, i.e., the number of tables associated with each word type.

An example of such a hierarchical model is the HDP bigram model of GGJ06, in which each word type w is associated with its own restaurant, where customers in that restaurant correspond to words that follow w in the corpus. All the bigram restaurants share a common base distribution P_1 over unigrams, which must be inferred. Predictions in this model are as follows:

$$P_2(w_i|\mathbf{h}_{-i}) = \frac{n_{(w_{i-1},w_i)}^{\mathbf{h}_{-i}} + \alpha_1 P_1(w_i|\mathbf{h}_{-i})}{n_{(w_{i-1},*)}^{\mathbf{h}_{-i}} + \alpha_1}$$

$$P_1(w_i|\mathbf{h}_{-i}) = \frac{t_{w_i}^{\mathbf{h}_{-i}} + \alpha_0 P_0(w_i)}{t_*^{\mathbf{h}_{-i}} + \alpha_0} \qquad (2)$$

where $\mathbf{h}_{-i} = (\mathbf{w}_{-i}, \mathbf{z}_{-i})$, $t_{w_i}^{\mathbf{h}_{-i}}$ is the number of tables labelled with w_i, and $t_*^{\mathbf{h}_{-i}}$ is the total number of occupied tables. Of particular note for our discussion is that in order to calculate these conditional distributions we must know the table assignments $\mathbf{z}_{-i}$ for each of the words in $\mathbf{w}_{-i}$. Moreover, in the Gibbs samplers often used for inference in these kinds of models, the counts are constantly changing over multiple samples, with tables going in and out of existence frequently. This can create significant bookkeeping issues in implementation, and motivated GGJ06 to present a method of computing approximate table counts based on word frequencies only.

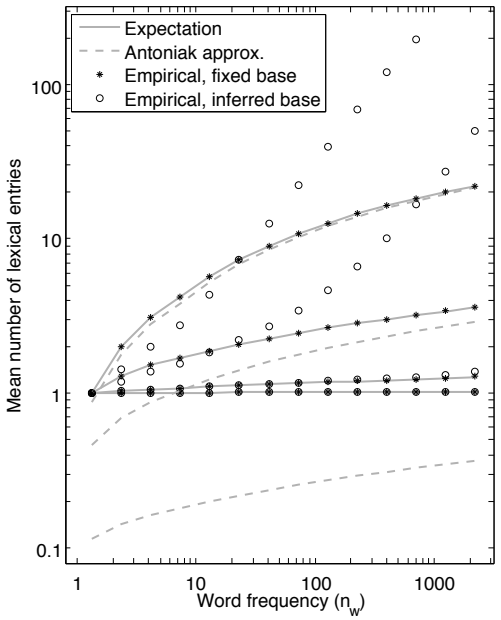


Figure 2. Comparison of several methods of approximating the number of tables occupied by words of different frequencies. For each method, results using $\alpha = \{100, 1000, 10000, 100000\}$ are shown (from bottom to top). Solid lines show the expected number of tables, computed using (3) and assuming P_1 is a fixed uniform distribution over a finite vocabulary (values computed using the Digamma formulation (7) are the same). Dashed lines show the values given by the Antoniak approximation (4) (the line for $\alpha = 100$ falls below the bottom of the graph). Stars show the mean of empirical table counts as computed over 1000 samples from an MCMC sampler in which P_1 is a fixed uniform distribution, as in the unigram LM. Circles show the mean of empirical table counts when P_1 is inferred, as in the bigram LM. Standard errors in both cases are no larger than the marker size. All plots are based on the 30114-word vocabulary and frequencies found in sections 0-20 of the WSJ corpus.

3 Approximating Table Counts

Rather than explicitly tracking the number of tables t_w associated with each word w in their bigram model, GGJ06 approximate the table counts using the expectation $E[t_w]$. Expected counts are used in place of $t_{w_i}^{\mathbf{h}_{-i}}$ and $t_*^{\mathbf{h}_{-i}}$ in (2). The exact expectation, due to Antoniak (1974), is

$$E[t_w] = \alpha_1 P_1(w) \sum_{i=1}^{n_w} \frac{1}{\alpha_1 P_1(w) + i - 1} \qquad (3)$$

Antoniak also gives an approximation to this expectation:

$$E[t_w] \approx \alpha_1 P_1(w) \log \frac{n_w + \alpha_1 P_1(w)}{\alpha_1 P_1(w)} \quad (4)$$

but provides no derivation. Due to a misinterpretation of Antoniak (1974), GGJ06 use an approximation that leaves out all the $P_1(w)$ terms from (4).[1] Figure 2 compares the approximation to the exact expectation when the base distribution is fixed. The approximation is fairly good when $\alpha P_1(w) > 1$ (the scenario assumed by Antoniak); however, in most NLP applications, $\alpha P_1(w) < 1$ in order to effect a sparse prior. (We return to the case of non-fixed based distributions in a moment.) As an extreme case of the paucity of this approximation consider $\alpha_1 P_1(w) = 1$ and $n_w = 1$ (i.e. only one customer has entered the restaurant): clearly $E[t_w]$ should equal 1, but the approximation gives $log(2)$.

We now provide a derivation for (4), which will allow us to obtain an $\mathcal{O}(1)$ formula for the expectation in (3). First, we rewrite the summation in (3) as a difference of fractional harmonic numbers:[2]

$$H_{(\alpha_1 P_1(w)+n_w-1)} - H_{(\alpha_1 P_1(w)-1)} \quad (5)$$

Using the recurrence for harmonic numbers:

$$E[t_w] \approx \alpha_1 P_1(w) \Big[H_{(\alpha_1 P_1(w)+n_w)} - \frac{1}{\alpha_1 P_1(w) + n_w} - H_{(\alpha_1 P_1(w)+n_w)} + \frac{1}{\alpha_1 P_1(w)} \Big] \quad (6)$$

We then use the asymptotic expansion, $H_F \approx \log F + \gamma + \frac{1}{2F}$, omiting trailing terms which are $\mathcal{O}(F^{-2})$ and smaller powers of F:[3]

$$E[t_w] \approx \alpha_1 P_1(w) \log \frac{n_w+\alpha_1 P_1(w)}{\alpha_1 P_1(w)} + \frac{n_w}{2(\alpha_1 P_1(w)+n_w)}$$

Omitting the trailing term leads to the approximation in Antoniak (1974). However, we can obtain an exact formula for the expectation by utilising the relationship between the Digamma function and the harmonic numbers: $\psi(n) = H_{n-1} - \gamma$.[4] Thus we can rewrite (5) as:[5]

$$E[t_w] = \alpha_1 P_1(w) \cdot \big[\psi(\alpha_1 P_1(w) + n_w) - \psi(\alpha_1 P_1(w)) \big] \quad (7)$$

[1] The authors of GGJ06 realized this error, and current implementations of their models no longer use these approximations, instead tracking table counts explicitly.

[2] Fractional harmonic numbers between 0 and 1 are given by $H_F = \int_0^1 \frac{1-x^F}{1-x} dx$. All harmonic numbers follow the recurrence $H_F = H_{F-1} + \frac{1}{F}$.

[3] Here, γ is the Euler-Mascheroni constant.

[4] Accurate $\mathcal{O}(1)$ approximations of the Digamma function are readily available.

[5] (7) can be derived from (3) using: $\psi(x+1) - \psi(x) = \frac{1}{x}$.

Explicit table tracking:
customer(w_i) $\rightarrow$ table(z_i)
$\{a:1,\ b:1,\ c:2,\ d:2,\ e:3,\ f:4,\ g:5,\ h:5\}$
table(z_i) $\rightarrow$ label(ℓ)
$\{1:The,\ 2:cats,\ 3:cats,\ 4:meow,\ 5:cats\}$
Histogram:
word type $\rightarrow$ { table occupancy $\rightarrow$ frequency }
$\{ The:\{2:1\},\ cats:\{1:1,\ 2:2\},\ meow:\{1:1\} \}$

Figure 3. The explicit table tracking and histogram representations for Figure 1.

A significant caveat here is that the expected table counts given by (3) and (7) are only valid when the base distribution is a constant. However, in hierarchical models such as GGJ06's bigram model and HDP models, the base distribution is not constant and instead must be inferred. As can be seen in Figure 2, table counts can diverge considerably from the expectations based on fixed P_1 when P_1 is in fact not fixed. Thus, (7) can be viewed as an approximation in this case, but not necessarily an accurate one. Since knowing the table counts is only necessary for inference in hierarchical models, but the table counts cannot be approximated well by any of the formulas presented here, we must conclude that the best inference method is still to keep track of the actual table counts. The naive method of doing so is to store which table each customer in the restaurant is seated at, incrementing and decrementing these counts as needed during the sampling process. In the following section, we describe an alternative method that reduces the amount of memory necessary for implementing HDPs. This method is also appropriate for hierarchical Pitman-Yor processes, for which no closed-form approximations to the table counts have been proposed.

4 Efficient Implementation of HDPs

As we do not have an efficient expected table count approximation for hierarchical models we could fall back to explicitly tracking which table each customer that enters the restaurant sits at. However, here we describe a more compact representation for the state of the restaurant that doesn't require explicit table tracking.[6] Instead we maintain a histogram for each dish w_i of the frequency of a table having a particular number of customers. Figure 3 depicts the histogram and explicit representations for the CRP state in Figure 1.

Our alternative method of inference for hierarchical Bayesian models takes advantage of their

[6] Teh et al. (2006) also note that the exact table assignments for customers are not required for prediction.

Algorithm 1 A new customer enters the restaurant

1: w: word type
2: P_0^w: Base probability for w
3: HD_w: Seating Histogram for w
4: **procedure** INCREMENT(w, P_0^w, HD_w)
5: $p_{share} \leftarrow \frac{n_w^{\mathbf{w}-1}}{n_w^{\mathbf{w}-1}+\alpha_0}$ ▷ share an existing table
6: $p_{new} \leftarrow \frac{\alpha_0 \times P_0^w}{n_w^{\mathbf{w}-1}+\alpha_0}$ ▷ open a new table
7: $r \leftarrow random(0, p_{share} + p_{new})$
8: **if** $r < p_{new}$ or $n_w^{\mathbf{w}-1} = 0$ **then**
9: $\text{HD}_w[1] = \text{HD}_w[1] + 1$
10: **else**
▷ Sample from the histogram of customers at tables
11: $r \leftarrow random(0, n_w^{\mathbf{w}-1})$
12: **for** $c \in \text{HD}_w$ **do** ▷ c: customer count
13: $r = r - (c \times \text{HD}_w[c])$
14: **if** $r \leq 0$ **then**
15: $\text{HD}_w[c] = \text{HD}_w[c] + 1$
16: Break
17: $n_w^{\mathbf{w}} = n_w^{\mathbf{w}-1} + 1$ ▷ Update token count

Algorithm 2 A customer leaves the restaurant

1: w: word type
2: HD_w: Seating histogram for w
3: **procedure** DECREMENT(w, P_0^w, HD_w)
4: $r \leftarrow random(0, n_w^{\mathbf{w}})$
5: **for** $c \in \text{HD}_w$ **do** ▷ c: customer count
6: $r = r - (c \times \text{HD}_w[c])$
7: **if** $r \leq 0$ **then**
8: $\text{HD}_w[c] = \text{HD}_w[c] - 1$
9: **if** $c > 1$ **then**
10: $\text{HD}_w[c-1] = \text{HD}_w[c-1] + 1$
11: Break
12: $n_w^{\mathbf{w}} = n_w^{\mathbf{w}} - 1$ ▷ Update token count

exchangeability, which makes it unnecessary to know exactly which table each customer is seated at. The only important information is how many tables exist with different numbers of customers, and what their labels are. We simply maintain a histogram for each word type w, which stores, for each number of customers m, the number of tables labeled with w that have m customers. Figure 3 depicts the explicit representation and histogram for the CRP state in Figure 1.

Algorithms 1 and 2 describe the two operations required to maintain the state of a CRP.[7] When a customer enters the restaurant (Alogrithm 1)), we sample whether or not to open a new table. If not, we sample an old table proportional to the counts of how many customers are seated there and update the histogram. When a customer leaves the restaurant (Algorithm 2), we decrement one of the tables at random according to the number of customers seated there. By exchangeability, it doesn't actually matter which table the customer was "really" sitting at.

[7] A C++ template class that implements the algorithm presented is made available at: http://homepages.inf.ed.ac.uk/tcohn/

5 Conclusion

We've shown that the HDP approximation presented in GGJ06 contained errors and inappropriate assumptions such that it significantly diverges from the true expectations for the most common scenarios encountered in NLP. As such we emphasise that that formulation should not be used. Although (7) allows $E[t_w]$ to be calculated exactly for constant base distributions, for hierarchical models this is not valid and no accurate calculation of the expectations has been proposed. As a remedy we've presented an algorithm that efficiently implements the true HDP without the need for explicitly tracking customer to table assignments, while remaining simple to implement.

Acknowledgements

The authors would like to thank Tom Griffiths for providing the code used to produce Figure 2 and acknowledge the support of the EPSRC (Blunsom, grant EP/D074959/1; Cohn, grant GR/T04557/01).

References

D. Aldous. 1985. Exchangeability and related topics. In *École d'Été de Probabilitiés de Saint-Flour XIII 1983*, 1–198. Springer.

C. E. Antoniak. 1974. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174.

J. DeNero, A. Bouchard-Côté, D. Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 314–323, Honolulu, Hawaii. Association for Computational Linguistics.

S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230.

J. R. Finkel, T. Grenager, C. D. Manning. 2007. The infinite tree. In *Proc. of the 45th Annual Meeting of the ACL (ACL-2007)*, Prague, Czech Republic.

S. Goldwater, T. Griffiths, M. Johnson. 2006a. Contextual dependencies in unsupervised word segmentation. In *Proc. of the 44th Annual Meeting of the ACL and 21st International Conference on Computational Linguistics (COLING/ACL-2006)*, Sydney.

S. Goldwater, T. Griffiths, M. Johnson. 2006b. Interpolating between types and tokens by estimating power-law generators. In Y. Weiss, B. Schölkopf, J. Platt, eds., *Advances in Neural Information Processing Systems 18*, 459–466. MIT Press, Cambridge, MA.

P. Liang, S. Petrov, M. Jordan, D. Klein. 2007. The infinite PCFG using hierarchical Dirichlet processes. In *Proc. of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-2007)*, 688–697, Prague, Czech Republic.

Y. W. Teh, M. I. Jordan, M. J. Beal, D. M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

A Succinct N-gram Language Model

Taro Watanabe **Hajime Tsukada** **Hideki Isozaki**
NTT Communication Science Laboratories
2-4 Hikaridai Seika-cho Soraku-gun Kyoto 619-0237 Japan
{taro,tsukada,isozaki}@cslab.kecl.ntt.co.jp

Abstract

Efficient processing of tera-scale text data is an important research topic. This paper proposes *lossless* compression of N-gram language models based on LOUDS, a succinct data structure. LOUDS succinctly represents a trie with M nodes as a $2M + 1$ bit string. We compress it further for the N-gram language model structure. We also use '*variable length coding*' and '*block-wise compression*' to compress *values* associated with nodes. Experimental results for three large-scale N-gram compression tasks achieved a significant compression rate *without any loss*.

1 Introduction

There has been an increase in available N-gram data and a large amount of web-scaled N-gram data has been successfully deployed in statistical machine translation. However, we need either a machine with hundreds of gigabytes of memory or a large computer cluster to handle them.

Either *pruning* (Stolcke, 1998; Church et al., 2007) or *lossy randomizing approaches* (Talbot and Brants, 2008) may result in a compact representation for the application run-time. However, the lossy approaches may reduce accuracy, and tuning is necessary. A lossless approach is obviously better than a lossy one if other conditions are the same. In addtion, a lossless approach can easly combined with pruning. Therefore, lossless representation of N-gram is a key issue even for lossy approaches.

Raj and Whittaker (2003) showed a general N-gram language model structure and introduced a lossless algorithm that compressed a sorted integer vector by recursively shifting a certain number of bits and by emitting index-value inverted vectors. However, we need more compact representation.

In this work, we propose a succinct way to represent the N-gram language model structure based on LOUDS (Jacobson, 1989; Delpratt et al., 2006). It was first introduced by Jacobson (1989) and requires only a small space close to the information-theoretic lower bound. For an M node ordinal trie, its information-theoretical lower bound is $2M - O(\lg M)$ bits ($\lg(x) = \log_2(x)$)

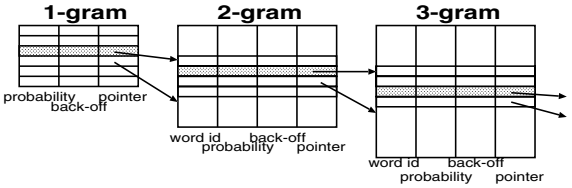


Figure 1: Data structure for language model

and LOUDS succinctly represents it by a $2M + 1$ bit string. The space is further reduced by considering the N-gram structure. We also use *variable length coding* and *block-wise compression* to compress the *values* associated with each node, such as word ids, probabilities or counts.

We experimented with English Web 1T 5-gram from LDC consisting of 25 GB of gzipped raw text N-gram counts. By using 8-bit floating point quantization [1], N-gram language models are compressed into 10 GB, which is comparable to a *lossy* representation (Talbot and Brants, 2008).

2 N-gram Language Model

We assume a back-off N-gram language model in which the conditional probability $Pr(w_n|w_1^{n-1})$ for an arbitrary N-gram $w_1^n = (w_1, ..., w_n)$ is recursively computed as follows.

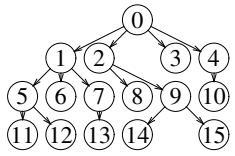
$$\begin{array}{ll} \alpha(w_1^n) & \text{if } w_1^n \text{ exists.} \\ \beta(w_1^{n-1})Pr(w_n|w_2^{n-1}) & \text{if } w_1^{n-1} \text{ exists.} \\ Pr(w_n|w_2^{n-1}) & \text{otherwise.} \end{array}$$

$\alpha(w_1^n)$ and $\beta(w_1^n)$ are smoothed probabilities and back-off coefficients, respectively.

The N-grams are stored in a trie structure as shown in Figure 1. N-grams of different orders are stored in different tables and each row corresponds to a particular w_1^n, consisting of a word id for w_n, $\alpha(w_1^n)$, $\beta(w_1^n)$ and a pointer to the first position of the succeeding $(n + 1)$-grams that share the same prefix w_1^n. The succeeding $(n+1)$-grams are stored in a contiguous region and sorted by the word id of w_{n+1}. The boundary of the region is determined by the pointer of the next N-gram in the

[1] The compact representation of the floating point is out of the scope of this paper. Therefore, we use the term *lossless* even when using floating point quantization.

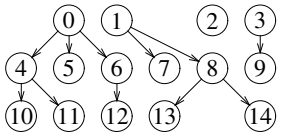
Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 341–344,
Suntec, Singapore, 4 August 2009.



(a) Trie structure

node id		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
bit position	0	1 2 3 4 5 6	7 8 9 10	11 12 13	14	15 16	17 18 19	20	21 22	23	24 25 26	27	28	29	30	31	32
LOUDS bit	**1**	**0 1 1 1 1 0**	**1 1 1 0**	**1 1 0**	**0**	**1 0**	**1 1 0**	**0**	**1 0**	**0**	**1 1 0**	**0**	**0**	**0**	**0**	**0**	**0**

(b) Corresponding LOUDS bit string



(c) Trie structure for N-gram

node id	0	1	2	3	4	5	6	7	8	9
bit position	0 1 2 3	4 5 6	7	8 9	10 11 12	13	14 15	16	17 18 19	20
LOUDS bit	**1 1 1 0**	**1 1 0**	**0**	**1 0**	**1 1 0**	**0**	**1 0**	**0**	**1 1 0**	**0**

(d) Corresponding N-gram optimized LOUDS bit string

Figure 2: Optimization of LOUDS bit string for N-gram data

row. When an N-gram is traversed, binary search is performed N times. If each word id corresponds to its node position in the unigram table, we can remove the word ids for the first order.

Our implementation merges across different orders of N-grams, then separates into multiple tables such as word ids, smoothed probabilities, back-off coefficients, and pointers. The starting positions of different orders are memorized to allow access to arbitrary orders. To store N-gram counts, we use three tables for word ids, counts and pointers. We share the same tables for word ids and pointers with additional probability and back-off coefficient tables.

To support distributed computation (Brants et al., 2007), we further split the N-gram data into "shards" by hash values of the first bigram. Unigram data are shared across shards for efficiency.

3 Succinct N-gram Structure

The table of pointers described in the previous section represents a trie. We use a succinct data structure LOUDS (Jacobson, 1989; Delpratt et al., 2006) for compact representation of the trie.

For an M node ordinal trie, there exist $\frac{1}{2M+1}\binom{2M+1}{M}$ different tries. Therefore, its information-theoretical lower bound is $\lg\left\lceil\frac{1}{2M+1}\binom{2M+1}{M}\right\rceil \approx 2M - O(\lg M)$ bits. LOUDS represents a trie with M nodes as a $2M + O(M)$ bit string.

The LOUDS bit string is constructed as follows. Starting from the root node, we traverse a trie in level order. For each node with $d \geq 0$ children, the bit string $\mathbf{1}^d\mathbf{0}$ is emitted. In addition, **10** is prefixed to the bit string emitted by an imaginary super-root node pointing to the root node. Figure 2(a) shows an example trie structure. The nodes are numbered in level order, and from left to right. The corresponding LOUDS bit string is shown in Figure 2(b). Since the root node 0 has four child nodes, it emits four **1**s followed by **0**, which marks the end of the node. Before the root node, we assume an imaginary super root node emits **10** for its only child, i.e., the root node. After the root node, its first child or node 1 follows. Since $(M+1)$**0**s and M**1**s are emitted for a trie with M nodes, LOUDS occupies $2M + 1$ bits.

We define a basic operation on the bit string. $\text{sel}_1(i)$ returns the position of the i-th **1**. We can also define similar operations over zero bit strings, $\text{sel}_0(i)$. Given sel_b, we define two operations for a node x. $\text{parent}(x)$ gives x's parent node and $\text{firstch}(x)$ gives x's first child node:

$$\text{parent}(x) = \text{sel}_1(x+1) - x - 1, \quad (1)$$
$$\text{firstch}(x) = \text{sel}_0(x+1) - x. \quad (2)$$

To test whether a child node exists, we simply check $\text{firstch}(x) \neq \text{firstch}(x+1)$. Similarly, the child node range is determined by $[\text{firstch}(x), \text{firstch}(x+1))$.

3.1 Optimizing N-gram Structure for Space

We propose removing redundant bits from the baseline LOUDS representation assuming N-gram structures. Since we do not store any information in the root node, we can safely remove the root so that the imaginary super-root node directly points to unigram nodes. The node ids are renumbered and the first unigram is 0. In this way, 2 bits are saved.

The N-gram data structure has a fixed depth N and takes a flat structure. Since the highest order N-grams have no child nodes, they emit $\mathbf{0}^{\mathcal{N}_N}$ in the tail of the bit stream, where $\mathcal{N}_n$ stands for the number of n-grams. By memorizing the starting position of the highest order N-grams, we can completely remove $\mathcal{N}_N$ bits.

The imaginary super-root emits $\mathbf{1}^{\mathcal{N}_1}\mathbf{0}$ at the beginning of the bit stream. By memorizing the bigram starting position, we can remove the $\mathcal{N}_1 + 1$ bits.

Finally, $\text{parent}(x)$ and $\text{firstch}(x)$ are rewritten as

integer seq.	52	156	260		364	
coding	0x34	0x9c	0x01	0x04	0x01	0x6c
boundary	**1**	**1**	**0**	**1**	**0**	**1**

Figure 3: Example of variable length coding

follows:

$$\text{parent}(x) = \text{sel}_1(x + 1 - \mathcal{N}_1) + \mathcal{N}_1 - x, \quad (3)$$

$$\text{firstch}(x) = \text{sel}_0(x) + \mathcal{N}_1 + 1 - x. \quad (4)$$

Figure 2(c) shows the N-gram optimized trie structure ($N = 3$) from Figure 2 with $\mathcal{N}_1 = 4$ and $\mathcal{N}_3 = 5$. The parent of node 8 is found by $\text{sel}_1(8{+}1{-}4) = 5$ and $5{+}4{-}8 = 1$. The first child is located by $\text{sel}_0(8) = 16$ and $16{+}4{+}1{-}8 = 13$.

When accessing the N-gram data structure, $\text{sel}_b(i)$ operations are used extensively. We use an auxiliary dictionary structure proposed by Kim et al. (2005) and Jacobson (1989) that supports an efficient $\text{sel}_1(i)$ ($\text{sel}_0(i)$) with the dictionary. We omit the details due to lack of space.

3.2 Variable Length Coding

The above method compactly represents pointers, but not associated *values*, such as word ids or counts. Raj and Whittaker (2003) proposed *integer compression* on each range of the word id sequence that shared the same N-gram prefix.

Here, we introduce a simple but more effective variable length coding for integer sequences of word ids and counts. The basic idea comes from encoding each integer by the smallest number of required bytes. Specifically, an integer within the range of 0 to 255 is coded as a 1-byte integer, the integers within the range of 256 to 65,535 are stored as 2-byte integers, and so on. We use an additional bit vector to indicate the boundary of the byte sequences. Figure 3 presents an example integer sequence, 52, 156, 260 and 364 with coded integers in hex decimals with boundary bits.

In spite of the length variability, the system can *directly access* a value at index i as bytes in $[\text{sel}_1(i) + 1, \text{sel}_1(i + 1) + 1)$ by the efficient sel_1 operation assuming that $\text{sel}_1(0)$ yields -1. For example, the value 260 at index 2 in Figure 3 is mapped onto the byte range of $[\text{sel}_1(2) + 1, \text{sel}_1(3) + 1) = [2, 4)$.

3.3 Block-wise Compression

We further compress every 8K-byte data block of all tables in N-grams by using a generic compression library, `zlib`, employed in UNIX `gzip`. We treat a sequence of 4-byte floats in the probability table as a byte stream, and compress every 8K-byte block. To facilitate random access to the compressed block, we keep track of the compressed block's starting offsets. Since the offsets are in sorted order, we can apply *sorted integer compression* (Raj and Whittaker, 2003). Since N-gram language model access preserves some locality, N-gram with block compression is still practical enough to be usable in our system.

4 Experiments

We applied the proposed representation to 5-gram trained by "*English Gigaword* 3rd Edition," "*English Web 1T* 5-gram" from LDC, and "*Japanese Web 1T* 7-gram" from GSK. Since their tendencies are the same, we only report in this paper the results on English Web 1T 5-gram, where the size of the count data in gzipped raw text format is 25GB, the number of N-grams is 3.8G, the vocabulary size is 13.6M words, and the number of the highest order N-grams is 1.2G.

We implemented an N-gram indexer/estimator using MPI inspired by the MapReduce implementation of N-gram language model indexing/estimation pipeline (Brants et al., 2007).

Table 1 summarizes the overall results. We show the initial indexed counts and the final language model size by differentiating compression strategies for the pointers, namely the 4-byte raw value (Trie), the sorted integer compression (Integer) and our succinct representation (Succinct). The "block" indicates block compression. For the sake of implementation simplicity, the sorted integer compression used a fixed 8-bit shift amount, although the original paper proposed recursively determined optimum shift amounts (Raj and Whittaker, 2003). 8-bit quantization was performed for probabilities and back-off coefficients using a simple binning approach (Federico and Cettolo, 2007).

N-gram counts were reduced from 23.59GB to 10.57GB by our succinct representation with block compression. N-gram language models of 42.65GB were compressed to 18.37GB. Finally, the 8-bit quantized N-gram language models are represented by 9.83GB of space.

Table 2 shows the compression ratio for the pointer table alone. Block compression employed on raw 4-byte pointers attained a large reduction that was almost comparable to sorted integer compression. Since large pointer value tables are sorted, even a generic compression algorithm could achieve better compression. Using our succinct representation, 2.4 bits are required for each N-gram. By using the "flat" trie structure, we approach closer to its information-theoretic lower bound beyond the LOUDS baseline. With block compression, we achieved 1.8 bits per N-gram.

Table 3 shows the effect of *variable length coding* and *block compression* for the word ids, counts, probabilities and back-off coefficients. After *variable-length coding*, the word id is almost half its original size. We assign a word id for each

		w/o block	w/ block
Counts	Trie	23.59 GB	12.21 GB
	Integer	14.59 GB	11.18 GB
	Succinct	12.62 GB	**10.57 GB**
Language model	Trie	42.65 GB	20.01 GB
	Integer	33.65 GB	18.98 GB
	Succinct	31.67 GB	**18.37 GB**
Quantized language model	Trie	24.73 GB	11.47 GB
	Integer	15.73 GB	10.44 GB
	Succinct	13.75 GB	**9.83 GB**

Table 1: Summary of N-gram compression

	total	per N-gram
4-byte Pointer	12.04 GB	27.24 bits
+block compression	2.42 GB	5.48 bits
Sorted Integer	3.04 GB	6.87 bits
+block compression	1.39 GB	3.15 bits
Succinct	1.06 GB	2.40 bits
+block compression	0.78 GB	1.76 bits

Table 2: Compression ratio for pointers

word according to its reverse sorted order of frequency. Therefore, highly frequent words are assigned smaller values, which in turn occupies less space in our variable length coding. With *block compression*, we achieved further 1 GB reduction in space. Since the word id sequence preserves local ordering for a certain range, even a generic compression algorithm is effective.

The most frequently observed count in N-gram data is *one*. Therefore, we can reduce the space by the variable length coding. Large compression rates are achieved for both probabilities and backoff coefficients.

5 Conclusion

We provided a succinct representation of the N-gram language model *without any loss*. Our method approaches closer to the information-theoretic lower bound beyond the LOUDS baseline. Experimental results showed our succinct representation drastically reduces the space for the pointers compared to the sorted integer compression approach. Furthermore, the space of N-grams was significantly reduced by *variable length coding* and *block compression*. A large amount of N-gram data is reduced from unindexed gzipped 25 GB text counts to 10 GB of indexed language models. Our representation is practical enough though we did not experimentally investigate the runtime efficiency in this paper. The proposed representation enables us to utilize a web-scaled N-gram in our MT competition system (Watanabe et al., 2008). Our succinct representation will encourage new research on web-scaled N-gram data without requiring a larger computer cluster or hundreds of gigabytes of memory.

	total	per N-gram
word id size (4 bytes)	14.09 GB	31.89 bits
+variable length	6.72 GB	15.20 bits
+block compression	5.57 GB	12.60 bits
count size (8 bytes)	28.28 GB	64.00 bits
+variable length	4.85 GB	10.96 bits
+block compression	4.22 GB	9.56 bits
probability size (4 bytes)	14.14 GB	32.00 bits
+block compression	9.55 GB	21.61 bits
8-bit quantization	3.54 GB	8.00 bits
+block compression	2.64 GB	5.97 bits
backoff size (4 bytes)	9.76 GB	22.08 bits
+block compression	2.48 GB	5.61 bits
8-bit quantization	2.44 GB	5.52 bits
+block compression	0.85 GB	1.92 bits

Table 3: Effects of block compression

Acknowledgments

We would like to thank Daisuke Okanohara for his open source implementation and extensive documentation of LOUDS, which helped our original coding.

References

T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. 2007. Large language models in machine translation. In *Proc. of EMNLP-CoNLL 2007*.

K. Church, T. Hart, and J. Gao. 2007. Compressing trigram language models with Golomb coding. In *Proc. of EMNLP-CoNLL 2007*.

O. Delpratt, N. Rahman, and R. Raman. 2006. Engineering the LOUDS succinct tree representation. In *Proc. of the 5th International Workshop on Experimental Algorithms*.

M. Federico and M. Cettolo. 2007. Efficient handling of n-gram language models for statistical machine translation. In *Proc. of the 2nd Workshop on Statistical Machine Translation*.

G. Jacobson. 1989. Space-efficient static trees and graphs. In *30th Annual Symposium on Foundations of Computer Science*, Nov.

D. K. Kim, J. C. Na, J. E. Kim, and K. Park. 2005. Efficient implementation of rank and select functions for succinct representation. In *Proc. of the 5th International Workshop on Experimental Algorithms*.

B. Raj and E. W. D. Whittaker. 2003. Lossless compression of language model structure and word identifiers. In *Proc. of ICASSP 2003*, volume 1.

A. Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proc. of the ARPA Workshop on Human Language Technology*.

D. Talbot and T. Brants. 2008. Randomized language models via perfect hash functions. In *Proc. of ACL-08: HLT*.

T. Watanabe, H. Tsukada, and H. Isozaki. 2008. NTT SMT system 2008 at NTCIR-7. In *Proc. of the 7th NTCIR Workshop*, pages 420–422.

Modeling Morphologically Rich Languages Using Split Words and Unstructured Dependencies

Deniz Yuret
Koç University
34450 Sariyer, Istanbul, Turkey
dyuret@ku.edu.tr

Ergun Biçici
Koç University
34450 Sariyer, Istanbul, Turkey
ebicici@ku.edu.tr

Abstract

We experiment with splitting words into their stem and suffix components for modeling morphologically rich languages. We show that using a morphological analyzer and disambiguator results in a significant perplexity reduction in Turkish. We present flexible n-gram models, FlexGrams, which assume that the $n-1$ tokens that determine the probability of a given token can be chosen anywhere in the sentence rather than the preceding $n-1$ positions. Our final model achieves 27% perplexity reduction compared to the standard n-gram model.

1 Introduction

Language models, i.e. models that assign probabilities to sequences of words, have been proven useful in a variety of applications including speech recognition and machine translation (Bahl et al., 1983; Brown et al., 1990). More recently, good results on lexical substitution and word sense disambiguation using language models have also been reported (Hawker, 2007; Yuret, 2007). Morphologically rich languages pose a challenge to standard modeling techniques because of their relatively large out-of-vocabulary rates and the regularities they possess at the sub-word level.

The standard n-gram language model ignores long-distance relationships between words and uses the independence assumption of a Markov chain of order $n-1$. Morphemes play an important role in the syntactic dependency structure in morphologically rich languages. The dependencies are not only between stems but also between stems and suffixes and if we use complete words as unit tokens, we will not be able to represent these sub-word dependencies. Our working hypothesis is that the performance of a language model is correlated by how much the probabilistic dependencies mirror the syntactic dependencies. We present flexible n-grams, FlexGrams, in which each token can be conditioned on tokens anywhere in the sentence, not just the preceding $n-1$ tokens. We also experiment with words split into their stem and suffix forms, and define stem-suffix FlexGrams where one set of offsets is applied to stems and another to suffixes. We evaluate the performance of these models on a morphologically rich language, Turkish.

2 The FlexGram Model

The FlexGram model relaxes the contextual assumption of n-grams and assumes that the $n-1$ tokens that determine the probability of a given token can be chosen anywhere in the sentence rather than at the preceding $n-1$ positions. This allows the ability to model long-distance relationships between tokens without a predefined left-to-right ordering and opens the possibility of using different dependency patterns for different token types.

Formal definition An order-n FlexGram model is specified by a tuple of dependency offsets $[d_1, d_2, \ldots, d_{n-1}]$ and decomposes the probability of a given sequence of tokens into a product of conditional probabilities for every token:

$$p(w_1, \ldots, w_k) = \prod_{w_i \in S} p(w_i | w_{i+d_1} \ldots w_{i+d_{n-1}})$$

The offsets can be positive or negative and the same set of offsets is applied to all tokens in the sequence. In order to represent a properly normalized probability model over the set of all finite length sequences, we check that the offsets of a FlexGram model does not result in a cycle. We show that using differing dependency offsets for stems and suffixes can improve the perplexity.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 345–348,
Suntec, Singapore, 4 August 2009.

3 Dataset

We used the Turkish newspaper corpus of Milliyet after removing sentences with 100 or more tokens. The dataset contains about 600 thousand sentences in the training set and 60 thousand sentences in the test set (giving a total of about 10 million words). The versions of the corpus we use developed by using different word-split strategies along with a sample sentence are explained below:

1. The *unsplit* dataset contains the raw corpus:
   `Kasparov bükemediği eli öpecek`
   (Kasparov is going to kiss the hand he cannot bend)
2. The *morfessor* dataset was prepared using the Morfessor (Creutz et al., 2007) algorithm:
   `Kasparov büke +mediği eli öp +ecek`
3. The *auto-split* dataset is obtained after using our unsupervised morphological splitter:
   `Kaspar +ov bük +emediği eli öp +ecek`
4. The *split* dataset contains words that are split into their stem and suffix forms by using a highly accurate supervised morphological analyzer (Yuret and Türe, 2006):
   `Kasparov bük +yAmA+dHk+sH el +sH öp +yAcAk`
5. The *split+0* version is derived from the *split* dataset by adding a zero-suffix to any stem that is not followed by a suffix:
   `Kasparov +0 bük +yAmA+dHk+sH el +sH öp +yAcAk`

Some statistics of the dataset are presented in Table 1. The vocabulary is taken to be the tokens that occur more than once in the training set and the OOV column shows the number of out-of-vocabulary tokens in the test set. The unique and 1-count columns give the number of unique tokens and the number of tokens that only occur once in the training set. Approximately 5% of the tokens in the *unsplit* test set are OOV tokens. In comparison, the ratio for a comparably sized English dataset is around 1%. Splitting the words into stems and suffixes brings the OOV ratio closer to that of English.

Model evaluation When comparing language models that tokenize data differently:

1. We take into account the true cost of the OOV tokens using a separate character-based model similar to Brown et al. (1992).
2. When reporting averages (perplexity, bits-per-word) we use a common denominator: the number of unsplit words.

Table 1: Dataset statistics (K for thousands, M for millions)

Dataset	Train	Test	OOV	Unique	1-count
unsplit	8.88M	0.91M	44.8K (4.94%)	430K	206K
morfessor	9.45M	0.98M	10.3K (1.05%)	167K	34.4K
auto-split	14.3M	1.46M	13.0K (0.89%)	128K	44.8K
split	12.8M	1.31M	17.1K (1.31%)	152K	75.4K
split+0	17.8M	1.81M	17.1K (0.94%)	152K	75.4K

4 Experiments

In this section we present a number of experiments that demonstrate that when modeling a morphologically rich language like Turkish, (i) splitting words into their stem and suffix forms is beneficial when the split is performed using a morphological analyzer and (ii) allowing the model to choose stem and suffix dependencies separately and flexibly results in a perplexity reduction, however the reduction does not offset the cost of zero suffixes. We used the SRILM toolkit (Stolcke, 2002) to simulate the behavior of FlexGram models by using count files as input. The interpolated Kneser-Ney smoothing was used in all our experiments.

Table 2: Total log probability (M for millions of bits).

	Split Dataset		Unsplit Dataset	
N	Word logp	OOV logp	Word logp	OOV logp
1	14.2M	0.81M	11.7M	2.32M
2	10.5M	0.64M	9.64M	1.85M
3	9.79M	0.56M	9.46M	1.59M
4	9.72M	0.53M	9.45M	1.38M
5	9.71M	0.51M	9.45M	1.25M
6	9.71M	0.50M	9.45M	1.19M

4.1 Using a morphological tagger and disambiguator

The *split* version of the corpus contains words that are split into their stem and suffix forms by using a previously developed morphological analyzer (Oflazer, 1994) and morphological disambiguator (Yuret and Türe, 2006). The analyzer produces all possible parses of a Turkish word using the two-level morphological paradigm and the disambiguator chooses the best parse based on the analysis of the context using decision lists. The integrated system was found to discover the correct morphological analysis for 96% of the words on a hand annotated out-of-sample test set. Table 2 gives the total log-probability (using $\log_2$) for the split and unsplit datasets using n-gram models of different order. We compute the perplexity of the two datasets using a common denominator: $2^{-\log_2(p)/N}$ where N=906,172 is taken to be the number of unsplit tokens. The best combination (order-6 word model combined with an order-9 letter model) gives a perplexity of 2,465 for the split dataset and 3,397 for the unsplit dataset,

which corresponds to a 27% improvement.

4.2 Separation of stem and suffix models

Only 45% of the words in the *split* dataset have suffixes. Each sentence in the *split+0* dataset has a regular *[stem suffix stem suffix ...]* structure. Table 3 gives the average cost of stems and suffixes in the two datasets for a regular 6-gram word model (ignoring the common OOV words). The log-probability spent on the zero suffixes in the split+0 dataset has to be spent on trying to decide whether to include a stem or suffix following a stem in the split dataset. As a result the difference in total log-probability between the two datasets is small (only 6% perplexity difference). The set of OOV tokens is the same for both the *split* and *split+0* datasets; therefore we ignore the cost of the OOV tokens as is the default SRILM behavior.

Table 3: Total log probability for the 6-gram word models on split and split+0 data.

	split dataset		split+0 dataset	
token type	number of tokens	total $-\log_2 p$	number of tokens	total $-\log_2 p$
stem	0.91M	7.80M	0.91M	7.72M
suffix	0.41M	1.89M	0.41M	1.84M
0-suffix	–	–	0.50M	0.21M
all	1.31M	9.69M	1.81M	9.78M

4.3 Using the FlexGram model

We perform a search over the space of dependency offsets using the *split+0* dataset and considered n-gram orders 2 to 6 and picked the dependency offsets within a window of $4n + 1$ tokens centered around the target. Table 4 gives the best models discovered for stems and suffixes separately and compares them to the corresponding regular n-gram models on the split+0 dataset. The numbers in parentheses give perplexity and significant reductions can be observed for each n-gram order.

Table 4: Regular ngram vs FlexGram models.

N	ngram-stem	ngram-suffix
2	-1 (1252)	-1 (5.69)
3	-2,-1 (418)	-2,-1 (5.29)
4	-3,-2,-1 (409)	-3,-2,-1 (4.79)
5	-4,-3,-2,-1 (365)	-4,-3,-2,-1 (4.80)
6	-5,-4,-3,-2,-1 (367)	-5,-4,-3,-2,-1 (4.79)

N	flexgram-stem	flexgram-suffix
2	-2 (596)	-1 (5.69)
3	+1,-2 (289)	+1,-1 (4.21)
4	+2,+1,-1 (189)	-2,+1,-1 (4.19)
5	+4,+2,+1,-1 (176)	-3,-2,+1,-1 (4.12)
6	+4,+3,+2,+1,-1 (172)	-4,-3,-2,+1,-1 (4.13)

However, some of these models cannot be used in combination because of cycles as we depict on the left side of Figure 1 for order 3. Table 5 gives the best combined models without cycles. We were able to exhaustively search all the patterns for orders 2 to 4 and we used beam search for orders 5 and 6. Each model is represented by its offset tuple and the resulting perplexity is given in parentheses. Compared to the regular n-gram models from Table 4 we see significant perplexity reductions up to order 4. The best order-3 stem-suffix FlexGram model can be seen on the right side of Figure 1.

Table 5: Best stem-suffix flexgram model combinations for the split+0 dataset.

N	flexgram-stem	flexgram-suffix	perplexity reduction
2	-2 (596)	-1 (5.69)	52.3%
3	-4,-2 (496)	+1,-1 (4.21)	5.58%
4	-4,-2,-1 (363)	-3,-2,-1 (4.79)	11.3%
5	-6,-4,-2,-1 (361)	-3,-2,-1 (4.79)	1.29%
6	-6,-4,-2,-1 (361)	-3,-2,-1 (4.79)	1.52%

5 Related work

Several approaches attempt to relax the rigid ordering enforced by the standard n-gram model. The skip-gram model (Siu and Ostendorf, Jan 2000) allows the skipping of one word within a given n-gram. Variable context length language modeling (Kneser, 1996) achieves a 10% perplexity reduction when compared to the trigrams by varying the order of the n-gram model based on the context. Dependency models (Rosenfeld, 2000) use the parsed dependency structure of sentences to build the language model as in grammatical trigrams (Lafferty et al., 1992), structured language models (Chelba and Jelinek, 2000), and dependency language models (Chelba et al., 1997). The dependency model governs the whole sentence and each word in a sentence is likely to have a different dependency structure whereas in our experiments with FlexGrams we use two connectivity patterns: one for stems and one for suffixes without the need for parsing.

6 Contributions

We have analyzed the effect of word splitting and unstructured dependencies on modeling Turkish, a morphologically complex language. Table 6 compares the models we have tested on our test corpus.

We find that splitting words into their stem and suffix components using a morphological analyzer and disambiguator results in significant perplexity reductions of up to 27%. FlexGram models outperform regular n-gram models (Tables 4 and 5)

Figure 1: Two FlexGram models where W represents a stem, s represents a suffix, and the arrows represent dependencies. The left model has stem offsets [+1,-2] and suffix offsets [+1,-1] and cannot be used as a directed graphical model because of the cycles. The right model has stem offsets [-4,-2] and suffix offsets [+1,-1] and is the best order-3 FlexGram model for Turkish.

Table 6: Perplexity for compared models.

N	unsplit	split	flexgram
2	3929	4360	5043
3	3421	2610	3083
4	3397	2487	2557
5	3397	2468	2539
6	3397	2465	2539

when using an alternating stem-suffix representation of the sentences; however Table 6 shows that the cost of the alternating stem-suffix representation (zero-suffixes) offsets this gain.

References

Lalit R. Bahl, Frederick Jelinek, and Robert L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190, 1983.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

Peter F. Brown et al. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18(1):31–40, 1992.

Ciprian Chelba and Frederick Jelinek. Recognition performance of a structured language model. *CoRR*, cs.CL/0001022, 2000.

Ciprian Chelba, David Engle, Frederick Jelinek, Victor M. Jimenez, Sanjeev Khudanpur, Lidia Mangu, Harry Printz, Eric Ristad, Ronald Rosenfeld, Andreas Stolcke, and Dekai Wu. Structure and performance of a dependency language model. In *Proc. Eurospeech '97*, pages 2775–2778, Rhodes, Greece, September 1997.

Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arisoy, Murat Saraclar, and Andreas Stolcke. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *TSLP*, 5 (1), 2007.

Tobias Hawker. USYD: WSD and lexical substitution using the Web1T corpus. In *SemEval-2007: 4th International Workshop on Semantic Evaluations*, 2007.

R. Kneser. Statistical language modeling using a variable context length. In *Proc. ICSLP '96*, volume 1, pages 494–497, Philadelphia, PA, October 1996.

John Lafferty, Daniel Sleator, and Davy Temperley. Grammatical trigrams: a probabilistic model of link grammar. In *AAAI Fall Symposium on Probabilistic Approaches to NLP*, 1992.

Kemal Oflazer. Two-level description of turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148, 1994.

Ronald Rosenfeld. Two decades of statistical language modeling: Where do we go from here. In *Proceedings of the IEEE*, volume 88, pages 1270–1278, 2000.

Manhung Siu and M. Ostendorf. Variable n-grams and extensions for conversational speech language modeling. *Speech and Audio Processing, IEEE Transactions on*, 8(1):63–75, Jan 2000. ISSN 1063-6676. doi: 10.1109/89.817454.

Andreas Stolcke. Srilm – an extensible language modeling toolkit. In *Proc. Int. Conf. Spoken Language Processing (ICSLP 2002)*, 2002.

Deniz Yuret. KU: Word sense disambiguation by substitution. In *SemEval-2007: 4th International Workshop on Semantic Evaluations*, June 2007.

Deniz Yuret and Ferhan Türe. Learning morphological disambiguation rules for turkish. In *HLT-NAACL 06*, June 2006.

Improved Smoothing for N-gram Language Models Based on Ordinary Counts

Robert C. Moore **Chris Quirk**
Microsoft Research
Redmond, WA 98052, USA
{bobmoore,chrisq}@microsoft.com

Abstract

Kneser-Ney (1995) smoothing and its variants are generally recognized as having the best perplexity of any known method for estimating N-gram language models. Kneser-Ney smoothing, however, requires nonstandard N-gram counts for the lower-order models used to smooth the highest-order model. For some applications, this makes Kneser-Ney smoothing inappropriate or inconvenient. In this paper, we introduce a new smoothing method based on ordinary counts that outperforms all of the previous ordinary-count methods we have tested, with the new method eliminating most of the gap between Kneser-Ney and those methods.

1 Introduction

Statistical language models are potentially useful for any language technology task that produces natural-language text as a final (or intermediate) output. In particular, they are extensively used in speech recognition and machine translation. Despite the criticism that they ignore the structure of natural language, simple N-gram models, which estimate the probability of each word in a text string based on the $N-1$ preceding words, remain the most widely used type of model.

The simplest possible N-gram model is the maximum likelihood estimate (MLE), which takes the probability of a word w_n, given the preceding context $w_1 \ldots w_{n-1}$, to be the ratio of the number of occurrences in a training corpus of the N-gram $w_1 \ldots w_n$ to the total number of occurrences of any word in the same context:

$$p(w_n|w_1 \ldots w_{n-1}) = \frac{C(w_1 \ldots w_n)}{\sum_{w'} C(w_1 \ldots w_{n-1} w')}$$

One obvious problem with this method is that it assigns a probability of zero to any N-gram that is not observed in the training corpus; hence, numerous smoothing methods have been invented that reduce the probabilities assigned to some or all observed N-grams, to provide a non-zero probability for N-grams not observed in the training corpus.

The best methods for smoothing N-gram language models all use a hierarchy of lower-order models to smooth the highest-order model. Thus, if $w_1 w_2 w_3 w_4 w_5$ was not observed in the training corpus, $p(w_5|w_1 w_2 w_3 w_4)$ is estimated based on $p(w_5|w_2 w_3 w_4)$, which is estimated based on $p(w_5|w_3 w_4)$ if $w_2 w_3 w_4 w_5$ was not observed, etc.

In most smoothing methods, the lower-order models, for all $N > 1$, are recursively estimated in the same way as the highest-order model. However, the smoothing method of Kneser and Ney (1995) and its variants are the most effective methods known (Chen and Goodman, 1998), and they use a different way of computing N-gram counts for all the lower-order models used for smoothing. For these lower-order models, the actual corpus counts $C(w_1 \ldots w_n)$ are replaced by

$$C'(w_1 \ldots w_n) = |\{w'|C(w' w_1 \ldots w_n) > 0\}|$$

In other words, the count used for a lower-order N-gram is the number of distinct word types that precede it in the training corpus.

The fact that the lower-order models are estimated differently from the highest-order model makes the use of Kneser-Ney (KN) smoothing awkward in some situations. For example, coarse-to-fine search using a sequence of lower-order to higher-order language models has been shown to be an efficient way of constraining high-dimensional search spaces for speech recognition (Murveit et al., 1993) and machine translation (Petrov et al., 2008). The lower-order models used in KN smoothing, however, are very poor estimates of the probabilities for N-grams that *have* been observed in the training corpus, so they are

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 349–352,
Suntec, Singapore, 4 August 2009.

$$p(w_n|w_1 \dots w_{n-1}) = \begin{cases} \alpha_{w_1 \dots w_{n-1}} \frac{C_n(w_1 \dots w_n) - D_{n,C_n(w_1 \dots w_n)}}{\sum_{w'} C_n(w_1 \dots w_{n-1} w')} & \\ + \beta_{w_1 \dots w_{n-1}} p(w_n|w_2 \dots w_{n-1}) & \text{if } C_n(w_1 \dots w_n) > 0 \\ \\ \gamma_{w_1 \dots w_{n-1}} p(w_n|w_2 \dots w_{n-1}) & \text{if } C_n(w_1 \dots w_n) = 0 \end{cases}$$

Figure 1: General language model smoothing schema

not suitable for use in coarse-to-fine search. Thus, two versions of every language model below the highest-order model would be needed to use KN smoothing in this case.

Another case in which use of special KN counts is problematic is the method presented by Nguyen et al. (2007) for building and applying language models trained on very large corpora (up to 40 billion words in their experiments). The scalability of their approach depends on a "backsorted trie", but this data structure does not support efficient computation of the special KN counts.

In this paper, we introduce a new smoothing method for language models based on ordinary counts. In our experiments, it outperformed all of the previous ordinary-count methods we tested, and it eliminated most of the gap between KN smoothing and the other previous methods.

2 Overview of Previous Methods

All the language model smoothing methods we will consider can be seen as instantiating the recursive schema presented in Figure 1, for all n such that $N \geq n \geq 2$,[1] where N is the greatest N-gram length used in the model.

In this schema, C_n denotes the counting method used for N-grams of length n. For most smoothing methods, C_n denotes actual training corpus counts for all n. For KN smoothing and its variants, however, C_n denotes actual corpus counts only when n is the greatest N-gram length used in the model, and otherwise denotes the special KN C' counts.

In this schema, each N-gram count is discounted according to a D parameter that depends, at most, on the N-gram length and the the N-gram count itself. The values of the α, β, and γ parameters depend on the context $w_1 \dots w_{n-1}$. For each context, the values of α, β, and γ must be set to produce a normalized conditional probability distribution. Additional constraints on the previous models we consider further reduce the degrees of freedom so that ultimately the values of these parameters are completely fixed by the values selected for the D parameters.

The previous smoothing methods we consider can be classified as either "pure backoff", or "pure interpolation". In pure backoff methods, all instances of $\alpha = 1$ and all instances of $\beta = 0$. The pure backoff methods we consider are Katz backoff and backoff absolute discounting, due to Ney et al.[2] In Katz backoff, if $C(w_1 \dots w_n)$ is greater than a threshold (here set to 5, as recommended by Katz) the corresponding $D = 0$; otherwise D is set according to the Good-Turing method.[3]

In backoff absolute discounting, the D parameters depends, at most, on n; there is either one discount per N-gram length, or a single discount used for all N-gram lengths. The values of D can be set either by empirical optimization on held-out data, or based on a theoretically optimal value derived from a leaving-one-out analysis, which Ney et al. show to be approximated for each N-gram length by $N_1/(N_1 + 2N_2)$, where N_r is the number of distinct N-grams of that length occuring r times in the training corpus.

In pure interpolation methods, for each context, β and γ are constrained to be equal. The models we consider that fall into this class are interpolated absolute discounting, interpolated KN, and modified interpolated KN. In these three methods, all instances of $\alpha = 1$.[4] In interpolated absolute discounting, the instances of D are set as in backoff absolute discounting. The same is true for inter-

[1] For $n = 2$, we take the expression $p(w_n|w_2 \dots w_{n-1})$ to denote a unigram probability estimate $p(w_2)$.

[2] For all previous smoothing methods other than KN, we refer the reader only to the excellent comparative study of smoothing methods by Chen and Goodman (1998). References to the original sources may be found there.

[3] Good-Turing discounting is usually expressed in terms of a discount ratio, but this can be reformulated as $D_r = r - d_r r$, where D_r is the subtractive discount for an N-gram occuring r times, and d_r is the corresponding discount ratio.

[4] Jelinek-Mercer smoothing would also be a pure interpolation instance of our language model schema, in which all instances of $D = 0$ and, for each context, $\alpha + \beta = 1$.

polated KN, but the lower-order models are estimated using the special KN counts.

In Chen and Goodman's (1998) modified interpolated KN, instead of one D parameter for each N-gram length, there are three: D_1 for N-grams whose count is 1, D_2 for N-grams whose count is 2, and D_3 for N-grams whose count is 3 or more. The values of these parameters may be set either by empirical optimization on held-out data, or by a theoretically-derived formula analogous to the Ney et al. formula for the one-discount case:

$$D_r = r - (r+1)Y\frac{N_{r+1}}{N_r},$$

for $1 \leq r \leq 3$, where $Y = N_1/(N_1 + 2N_2)$, the discount value derived by Ney et al.

3 The New Method

Our new smoothing method is motivated by the observation that unsmoothed MLE language models suffer from two somewhat independent sources of error in estimating probabilities for the N-grams observed in the training corpus. The problem that has received the most attention is the fact that, on the whole, the MLE probabilities for the observed N-grams are overestimated, since they end up with all the probability mass that should be assigned to the unobserved N-grams. The discounting used in Katz backoff is based on the Good-Turing estimate of exactly this error.

Another source of error in MLE models, however, is quantization error, due to the fact that only certain estimated probability values are possible for a given context, depending on the number of occurrences of the context in the training corpus. No pure backoff model addresses this source of error, since no matter how the discount parameters are set, the number of possible probability values for a given context cannot be increased just by discounting observed counts, as long as all N-grams with the same count receive the same discount. Interpolation models address quantization error by interpolation with lower-order estimates, which should have lower quantization error, due to higher context counts. As we have noted, most existing interpolation models are constrained so that the discount parameters fully determine the interpolation parameters. Thus the discount parameters have to correct for both types of error.[5]

Our new model provides additional degrees of freedom so the α and β interpolation parameters can be set independently of the discount parameters D, with the intention that the α and β parameters correct for quantization error, and the D parameters correct for overestimation error. This is accomplished by relaxing the link between the β and γ parameters. We require that for each context, $\alpha \geq 0$, $\beta \geq 0$, and $\alpha + \beta = 1$, and that for every $D_{n,C_n(w_1 \ldots w_n)}$ parameter, $0 \leq D \leq C_n(w_1 \ldots w_n)$. For each context, whatever values we choose for these parameters within these constraints, we are guaranteed to have some probability mass between 0 and 1 left over to be distributed across the unobserved N-grams by a unique value of γ that normalizes the conditional distribution.

Previous smoothing methods suggest several approaches to setting the D parameters in our new model. We try four such methods here:

1. The single theory-based discount for each N-gram length proposed by Ney et al.,
2. A single discount used for all N-gram lengths, optimized on held-out data,
3. The three theory-based discounts for each N-gram length proposed by Chen and Goodman,
4. A novel set of three theory-based discounts for each N-gram length, based on Good-Turing discounting.

The fourth method is similar to the third, but for the three D parameters per context, we use the discounts for 1-counts, 2-counts, and 3-counts estimated by the Good-Turing method. This yields the formula

$$D_r = r - (r+1)\frac{N_{r+1}}{N_r},$$

which is identical to the Chen-Goodman formula, except that the Y factor is omitted. Since Y is generally between 0 and 1, the resulting discounts will be smaller than with the Chen-Goodman formula.

To set the α and β parameters, we assume that there is a single unknown probability distribution for the amount of quantization error in every N-gram count. If so, the total quantization error for a given context will tend to be proportional to the

[5] Jelinek-Mercer smoothing is an exception to this generalization, but since it has only interpolation parameters and no discount parameters, it forces the interpolation parameters to do the same double duty that other models force the discount parameters to do.

number of distinct counts for that context, in other words, the number of distinct word types occurring in that context. We then set α and β to replace the proportion of the total probability mass for the context represented by the estimated quantization error with probability estimates derived from the lower-order models:

$$\beta_{w_1 \ldots w_{n-1}} = \delta \frac{|\{w' | C_n(w_1 \ldots w_{n-1} w') > 0\}|}{\sum_{w'} C_n(w_1 \ldots w_{n-1} w')}$$
$$\alpha_{w_1 \ldots w_{n-1}} = 1 - \beta_{w_1 \ldots w_{n-1}}$$

where δ is the estimated mean of the quantization error introduced by each N-gram count.

We use a single value of δ for all contexts and all N-gram lengths. As an *a priori* "theory"-based estimate, we assume that, since the distance between possible N-gram counts, after discounting, is approximately 1.0, their mean quantization error would be approximately 0.5. We also try setting δ by optimization on held-out data.

4 Evaluation and Conclusions

We trained and measured the perplexity of 4-gram language models using English data from the WMT-06 Europarl corpus (Koehn and Monz, 2006). We took 1,003,349 sentences (27,493,499 words) for training, and 2000 sentences each for testing and parameter optimization.

We built models based on six previous approaches: (1) Katz backoff, (2) interpolated absolute discounting with Ney et al. formula discounts, backoff absolute discounting with (3) Ney et al. formula discounts and with (4) one empirically optimized discount, (5) modified interpolated KN with Chen-Goodman formula discounts, and (6) interpolated KN with one empirically optimized discount. We built models based on four ways of computing the D parameters of our new model, with a fixed $\delta = 0.5$: (7) Ney et al. formula discounts, (8) one empirically optimized discount, (9) Chen-Goodman formula discounts, and (10) Good-Turing formula discounts. We also built a model (11) based on one empirically optimized discount $D = 0.55$ and an empircially optimized value of $\delta = 0.9$. Table 1 shows that each of these variants of our method had better perplexity than every previous ordinary-count method tested.

Finally, we performed one more experiment, to see if the best variant of our model (11) combined with KN counts would outperform either variant of interpolated KN. It did not, yielding a perplexity of 53.9 after reoptimizing the two free parameters of the model with the KN counts. However, the best variant of our model eliminated 65% of the difference in perplexity between the best previous ordinary-count method tested and the best variant of KN smoothing tested, suggesting that it may currently be the best approach when language models based on ordinary counts are desired.

	Method	PP
1	Katz backoff	59.8
2	interp-AD-fix	62.6
3	backoff-AD-fix	59.9
4	backoff-AD-opt	58.8
5	KN-mod-fix	52.8
6	KN-opt	53.0
7	new-AD-fix	56.3
8	new-AD-opt	55.6
9	new-CG-fix	57.4
10	new-GT-fix	56.1
11	new-AD-2-opt	54.9

Table 1: 4-gram perplexity results

References

Chen, Stanley F., and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University.

Kneser, Reinhard, and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of ICASSP-95*, vol. 1, 181–184.

Koehn, Philipp, and Christof Monz. 2006. Manual and automatic evaluation of machine translation between European languages. In *Proceedings of WMT-06*, 102–121.

Murveit, Hy, John Butzberger, Vassilios Digalakis, and Mitch Weintraub. 1993. Progressive search algorithms for large-vocabulary speech recognition. In *Proceedings of HLT-93*, 87–90.

Nguyen, Patrick, Jianfeng Gao, and Milind Mahajan. 2007. MSRLM: a scalable language modeling toolkit. Technical Report MSR-TR-2007-144. Microsoft Research.

Petrov, Slav, Aria Haghighi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of ACL-08*. 108–116.

Updating a Name Tagger Using Contemporary Unlabeled Data

Cristina Mota
L2F (INESC-ID) & IST & NYU
Rua Alves Redol 9
1000-029 Lisboa Portugal
cmota@ist.utl.pt

Ralph Grishman
New York University
Computer Science Department
New York NY 10003 USA
grishman@cs.nyu.edu

Abstract

For many NLP tasks, including named entity tagging, semi-supervised learning has been proposed as a reasonable alternative to methods that require annotating large amounts of training data. In this paper, we address the problem of analyzing new data given a semi-supervised NE tagger trained on data from an earlier time period. We will show that updating the unlabeled data is sufficient to maintain quality over time, and outperforms updating the labeled data. Furthermore, we will also show that augmenting the unlabeled data with older data in most cases does not result in better performance than simply using a smaller amount of current unlabeled data.

1 Introduction

Brill (2003) observed large gains in performance for different NLP tasks solely by increasing the size of unlabeled data, but stressed that for other NLP tasks, such as named entity recognition (NER), we still need to focus on developing tools that help to increase the size of annotated data.

This problem is particularly crucial when processing languages, such as Portuguese, for which the labeled data is scarce. For instance, in the first NER evaluation for Portuguese, HAREM (Santos and Cardoso, 2007), only two out of the nine participants presented systems based on machine learning, and they both argued they could have achieved significantly better results if they had larger training sets.

Semi-supervised methods are commonly chosen as an alternative to overcome the lack of annotated resources, because they present a good trade-off between amount of labeled data needed and performance achieved. Co-training is one of those methods, and has been extensively studied in NLP (Nigam and Ghani, 2000; Pierce and Cardie, 2001; Ng and Cardie, 2003; Mota and Grishman, 2008). In particular, we showed that the performance of a name tagger based on co-training decays as the time gap between training data (seeds and unlabeled data) and test data increases (Mota and Grishman, 2008). Compared to the original classifier of Collins and Singer (1999) that uses seven seeds, we used substantially larger seed sets (more than 1000), which raises the question of which of the parameters (seeds or unlabeled data) are causing the performance deterioration.

In the present study, we investigated two main questions, from the point of view of a developer who wants to analyze a new data set, given an NE tagger trained with older data. First, we studied whether it was better to update the seeds or the unlabeled data; then, we analyzed whether using a smaller amount of current unlabeled data could be better than increasing the amount of unlabeled data drawn from older sources. The experiments show that using contemporary unlabeled data is the best choice, outperforming most experiments with larger amounts of older unlabeled data and all experiments with contemporary seeds.

2 Contemporary labeled data in NLP

The speech community has been defending for some time now the idea of having similar temporal data for training and testing automatic speech recognition systems for broadcast news. Most works focus on improving out-of-vocabulary (OOV) rates, to which new names contribute significantly. For instance, Palmer and Ostendorf (2005) aiming at reducing the error rate due to OOV names propose to generate offline name lists from diverse sources, including temporally relevant news texts; Federico and Bertoldi (2004), and Martins et al. (2006) propose to daily adapt the statistical language model of a broadcast

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 353–356,
Suntec, Singapore, 4 August 2009.

news transcription system, exploiting contemporary newswire texts available on the web; Auzanne et al. (2000) proposed a time-adaptive language model, studying its impact over a period of five months on the reduction of OOV rate, word error rate and retrieval accuracy on a spoken document retrieval system.

Concerning variations over longer periods of time, we observed that the performance of a semi-supervised name tagger decays over a period of eight years, which seems to be directly related with the fact that the texts used to train and test the tagger also show a tendency to become less similar over time (Mota and Grishman, 2008); Batista et al. (2008) also observed a decaying tendency in the performance of a system for recovering capitalization over a period of six years, proposing to retrain a MaxEnt model using additional contemporary written texts.

3 Name tagger overview

We assessed the name tagger described in Mota and Grishman (2008) to recognize names of people, organizations and locations. The tagger is based on the co-training NE classifier proposed by Collins and Singer (1999), and is comprised of several components organized sequentially (cf. Figure 1).

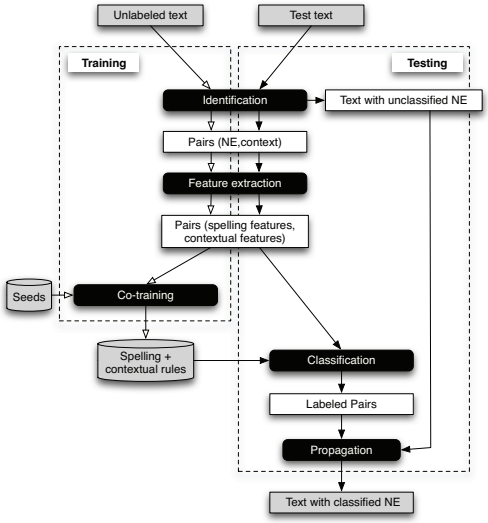


Figure 1: NE tagger architecture

4 Data sets

CETEMPúblico (Rocha and Santos, 2000) is a Portuguese journalistic corpus with 180 million words that spans eight years of news, from 1991 to 1998. The minimum size of epoch (time span of data set) available for analysis is a six-month period, corresponding either to the first half of the year or the second.

The data sets were created using the first 8256 extracts[1] within each six-month period of the politics section of the corpus: the first 192 are used to collect seeds, the next 208 extracts are used as test sets and the remaining 7856 are used to collect the unlabeled examples. The seeds correspond to the first 1150 names occurring in those extracts. From the list of unlabeled examples obtained after the NE identification stage, only the first 41226 examples of each epoch were used to bootstrap in the classification stage.

5 Experiments

We denote by S, U and T, respectively, the seed, unlabeled and test texts, and by (S_i, U_j, T_k) a training-test configuration, where $91a \leq i, j, k \leq 98b$, i.e., epochs i, j and k vary between the first half of 1991 (91a) and the second half of 1998 (98b). For instance, the training-test configuration ($S_{i=91a...98b}$, $U_{i=91a...98b}$, $T_{j=98b}$) represents the training-test configuration where the test set was drawn from epoch 98b, and the tagger was trained in turn with seeds and unlabeled data drawn from the same epoch i that varied from 91a to 98b.

5.1 Do we need contemporary labeled data?

In order to understand whether it is better to label examples falling within the epoch of the test set or to keep using old labeled data while bootstrapping with contemporary unlabeled data, we fixed the test set to be within the last epoch of the interval (98b), and performed backward experiments, i.e., we varied the epoch of either the seeds or the unlabeled data backwards. The choice of fixing the test within the last epoch of the interval is the one that most approximates a real situation where one has a tagger trained with old data and wants to process a more recent text.

Figure 2 shows the results for both experiments, where ($S_{j=98b}$, $U_{i=91a...98b}$, $T_{j=98b}$) represents the experiment where the test was within the same epoch as the seeds and the unlabeled data were drawn from a single, variable, epoch in turn, and ($S_{i=91a...98b}$, $U_{j=98b}$, $T_{j=98b}$) represents the experiment where the test was within the epoch of the

[1] Extracts are typically two paragraphs.

unlabeled data and the seeds were drawn in turn from each of the epochs; the graphic also shows the baseline backward training (varying the epoch of both the seeds and the unlabeled data together).

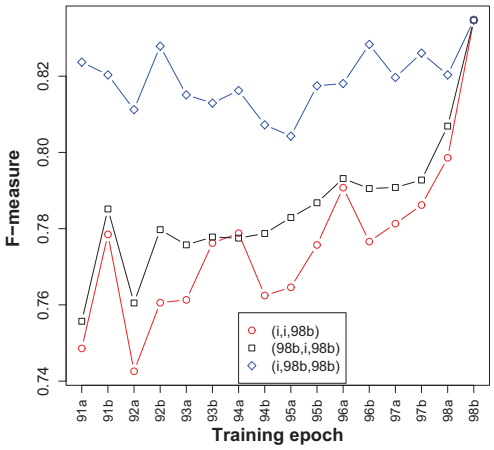


Figure 2: F-measure over time for test set $98b$ with configurations: ($S_{i=91a...98b}$, $U_{i=91a...98b}$, $T_{j=98b}$), ($S_{j=98b}$, $U_{i=91a...98b}$, $T_{j=98b}$), and ($S_{i=91a...98b}$, $U_{j=98b}$, $T_{j=98b}$)

As can be seen, there is a small gain in performance by using seeds within the epoch of the test set, but the decay is still observable as we increase the time gap between the unlabeled data and the test set. On the contrary, if we use unlabeled data within the epoch of the test set, we hardly see a degradation trend as the time gap between the epochs of seeds and test set is increased.

An examination of the results shows that, for instance, *Sendero Luminoso* received the correct classification of organization when the tagger is trained with unlabeled data drawn from the same epoch, but is incorretly classified as person when trained with data that is not contemporary with the test set. Even though that name is not a seed in any of the cases, it occurs twice in good contexts for organization in unlabeled data contemporary with the test set (*líder do Sendero Luminoso*/*leader of the Shining Path* and *acções do Sendero Luminoso*/actions of the Shining Path), while it does not occur in the unlabeled data that is not contemporary. Given that both the name spelling and the context in the test set, *o messianismo do peruano Sendero Luminoso*/*the messianism of the Peruvian Shining Path*, are insufficient to assign a correct label, the occurrence of the name in the contemporary unlabeled data contributes to its correct classification in the test set.

5.2 Is more older unlabeled data better?

The second question we addressed was whether having more older unlabeled data could result in better performance than less data but within the epoch of the test set. In this case, we conducted two backward experiments, augmenting the unlabeled data backwards with older data than the test set (98b), starting in the previous epoch (98a): in the first experiment, the seeds were within the same epoch as the test set, and in the second experiment the seeds were within the same epoch as the unlabeled set being added. This corresponds to configurations ($S_{j=98b}$, $U'_{i=91a...98a}$, $T_{j=98b}$) and ($S_{i=91a...98a}$, $U'_{i=91a...98a}$, $T_{j=98b}$), respectively, where $U'_i = \bigcup_{k=i}^{98a} U_k$.

In Figure 3, we show the result of these configurations together with the result of the backward experiment corresponding to configuration ($S_{i=91a...98b}$, $U_{j=98b}$, $T_{j=98b}$), also represented in Figure 2. We note that, in the case of the former experiments, the size of the unlabeled examples is increasing in the direction 98a to 91a.

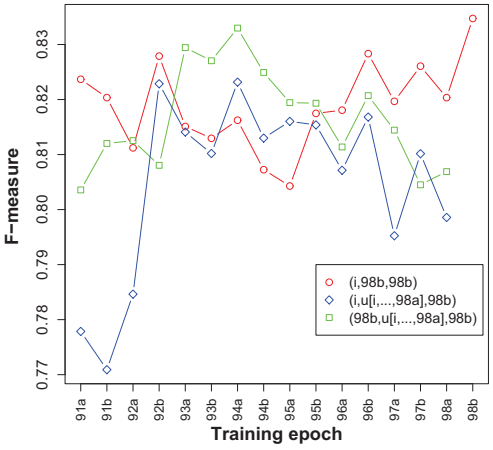


Figure 3: F-measure for test set 98b with configurations ($S_{i=91a...98b}$, $U_{j=98b}$, $T_{j=98b}$), ($S_{j=98b}$, $U'_{i=91a...98a}$, $T_{j=98b}$) and ($S_{i=91a...98a}$, $U'_{i=91a...98a}$, $T_{j=98b}$), where $U'_i = \bigcup_{k=i}^{98a} U_k$

As can be observed, increasing the size of the unlabeled data does not necessarily result in better performance: for both choices of seeds, performance sometimes improves, sometimes worsens, as the unlabeled data grows (following the curves

from right to left).

Furthermore, the tagger trained with more unlabeled data in most cases did not outperform the tagger trained with less unlabeled data selected from the epoch of the test set.

6 Discussion and future directions

We conducted experiments varying the epoch of seeds and unlabeled data of a named entity tagger based on co-training. We observed that the performance decay resulting from increasing the time gap between training data (seeds and unlabeled examples) and the test set can be slightly attenuated by using the seeds contemporary with the test set. The gain is larger if one uses older seeds and contemporary unlabeled data, a strategy that, in most of the experiments, results in better performance than using increasing sizes of older unlabeled data.

These results suggest that we may not need to label new data nor train our tagger with increasing sizes of data, as long as we are able to train it with unlabeled data time compatible with the test set.

In the future, one issue that needs clarification is why bootstraping from contemporary labeled data had so little influence on the performance of co-training, and if other semi-supervised approches are also sensitive to this question.

Acknowledgment

The first author's research work was funded by Fundação para a Ciência e a Tecnologia through a doctoral scholarship (ref.: SFRH/BD/3237/2000).

References

Cédric Auzanne, John S. Garofolo, Jonathan G. Fiscus, and William M. Fisher. 2000. Automatic language model adaptation for spoken document retrieval. In *Proceedings of RIAO 2000 Conference on Content-Based Multimedia Information Access*.

Fernando Batista, Nuno Mamede, and Isabel Trancoso. 2008. Language dynamics and capitalization using maximum entropy. In *Proceedings of ACL-08: HLT, Short Papers*, pages 1–4, Columbus, Ohio, June. Association for Computational Linguistics.

Eric Brill. 2003. Processing natural language without natural language processing. In *CICLing*, pages 360–369.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on EMNLP*.

Marcello Federico and Nicola Bertoldi. 2004. Broadcast news lm adaptation over time. *Computer Speech & Language*, 18(4):417–435.

Ciro Martins, António Teixeira, and João Neto. 2006. Dynamic vocabulary adaptation for a daily and real-time broadcast news transcription system. In *IEEE/ACL Workshop on Spoken Language Technology*, Aruba.

Cristina Mota and Ralph Grishman. 2008. Is this NE tagger getting old? In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may.

Vincente Ng and Claire Cardie. 2003. Weakly supervised natural language learning without redundant views. In *NAACL'03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 94–101, Morristown, NJ, USA. ACL.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of CIKM*, pages 86–93.

David D. Palmer and Mari Ostendorf. 2005. Improving out-of-vocabulary name resolution. *Computer Speech & Language*, 19(1):107–128.

David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP-2001)*.

Paulo Rocha and Diana Santos. 2000. Cetempúblico: Um corpus de grandes dimensões de linguagem jornalística portuguesa. In Maria das Graças Volpe Nunes, editor, *Actas do V Encontro para o processamento computacional da língua portuguesa escrita e falada PROPOR 2000*, pages 131–140, Atibaia, São Paulo, Brasil.

Diana Santos and Nuno Cardoso, editors. 2007. *Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área*. Linguateca, 12 de Novembro.

Arabic Cross-Document Coreference Detection

Asad Sayeed,[1,2] **Tamer Elsayed,**[1,2] **Nikesh Garera,**[1,6] **David Alexander,**[1,3]
Tan Xu,[1,4] **Douglas W. Oard,**[1,4,5] **David Yarowsky,**[1,6] **Christine Piatko**[1]
[1]Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, MD, USA—[2]Dept. of Computer Science, University of Maryland, College Park, MD, USA—[3]BBN Technologies, Cambridge, MA, USA—[4]College of Information Studies, University of Maryland, College Park, MD, USA—[5]UMIACS, University of Maryland, College Park, MD, USA—[6]Dept. of Computer Science, Johns Hopkins University, Baltimore, MD, USA
{asayeed,telsayed}@cs.umd.edu, ngarera@cs.jhu.edu, dalexand@bbn.com,
{tanx,oard}@umd.edu, yarowsky@cs.jhu.edu, Christine.Piatko@jhuapl.edu

Abstract

We describe a set of techniques for Arabic cross-document coreference resolution. We compare a baseline system of exact mention string-matching to ones that include local mention context information as well as information from an existing machine translation system. It turns out that the machine translation-based technique outperforms the baseline, but local entity context similarity does not. This helps to point the way for future cross-document coreference work in languages with few existing resources for the task.

1 Introduction

Our world contains at least two noteworthy George Bushes: President George H. W. Bush and President George W. Bush. They are both frequently referred to as "George Bush." If we wish to use a search engine to find documents about one of them, we are likely also to find documents about the other. Improving our ability to find *all* documents referring to one and *none* referring to the other in a targeted search is a goal of cross-document entity coreference detection. Here we describe some results from a system we built to perform this task on Arabic documents. We base our work partly on previous work done by Bagga and Baldwin (Bagga and Baldwin, 1998), which has also been used in later work (Chen and Martin, 2007). Other work such as Lloyd et al. (Lloyd, 2006) focus on techniques specific to English.

The main contribution of this work to cross-document coreference lies in the conditions under which it was done. Even now, there is no large-scale resource—in terms of annotated data—for cross-document coreference in Arabic as there is in English (e.g. WebPeople (Artiles, 2008)). Thus, we employed techniques for high-performance processing in a resource-poor environment. We provide early steps in cross-document coreference detection for resource-poor languages.

2 Approach

We treat cross-document entities as a set of graphs consisting of links between within-document entities. The graphs are disjoint. Each of our systems produces a list of such links as within-document entity pairs (A, B). We obtain within-document entities by running the corpus through a within-document coreference resolver—in this case, Serif from BBN Technologies.

To create the entity clusters, we use a union-find algorithm over the pairs. If links (A, B) and (C, B) appear in the system output, then $\{A, B, C\}$ are one entity. Similarly, if (X, Y) and (Z, Y) appear in the output, then it will find that $\{X, Y, Z\}$ are one entity. If the algorithm later discovers link (B, Z) in the system output, it will decide that $\{A, B, C, X, Y, Z\}$ are an entity. This is efficiently implemented via a hash table whose keys and values are both within-document entity IDs, allowing the implementation of easily-searched linked lists.

2.1 The baseline system

The baseline system uses a string matching criterion to determine whether two within-document entities are similar enough to be considered as part of the same cross-document entity. Given within-document entities A and B, the criterion is implemented as follows:

1. Find the mention strings $\{a_1, a_2, \ldots\}$ and

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 357–360,
Suntec, Singapore, 4 August 2009.

$\{b_1, b_2, \ldots\}$ of A and B, respectively that are the longest for that within-document entity in the given document. (There may be more than one longest mention of equal length for a given entity.)

2. If *any* longest mention strings a_n and b_m exist such that $a_n = b_m$ (exact string match), then A and B are considered to be part of the same cross-document entity. Otherwise, they are considered to be different entities.

When the system decides that two within-document entities are connected as a single cross-document entity, it emits a link between within-document entities A and B represented as the pair (A, B). We maintain a list of such links, but we omit all links between within-document entities in the same document.

The output of the system is a list of pairwise links. The following two experimental systems also produce lists of pairwise links. Union is performed between the baseline system's list and the lists produced by the other systems to create lists of pairs that include the information in the baseline. However, each of the following systems' outputs are merged *separately* with the baseline. By including the baseline results in each system, we are able to clarify the potential of each additional technique to improve performance over a technique that is cheap to run under any circumstances, especially given that our experiments are focused on *increasing* the number of links in an Arabic context where links are likely to be disrupted by spelling variations.

2.2 Translingual projection

We implement a novel cross-language approach for Arabic coreference resolution by expanding the space of exact match comparisons to approximate matches of English translations of the Arabic strings. The intuition for this approach is that often the Arabic strings of the same named entity may differ due to misspellings, titles, or aliases that can be corrected in the English space. The English translations were obtained using a standard statistical machine translation system (Chiang, 2007; Li, 2008) and then compared using an alias match.

The algorithm below describes the approach, applied to any Arabic named entities that fail the baseline string-match test:

1. For a given candidate Arabic named entity pair (A, B), we project them into English by translating the mentions using a standard statistical machine translation toolkit. Using the projected English pair, say, (A', B') we perform the following tests to determine whether A and B are co-referent:
   (a) We do an exact string-match test in the English space using the projected entities (A', B'). The exact string match test is done exactly as in the baseline system, using the set of longest named entities in their respective co-reference chains.
   (b) If (A', B') fail in the exact string-match test as in the baseline, then we test whether they belong to a list of high confidence co-referent named-entity pairs[1] precomputed for English using alias-lists derived from Wikipedia.
   (c) If (A', B') fails (a) and (b) then (A, B) is deemed as non-coreferent.

While we hypothesize that translingual projection via English should help in increasing recall since it can work with non-exact string matches, it may also help in increasing precision based on the assumption that a name of American or English origin might have different variants in Arabic and that translating to English can help in merging those variants, as shown in figure 1.

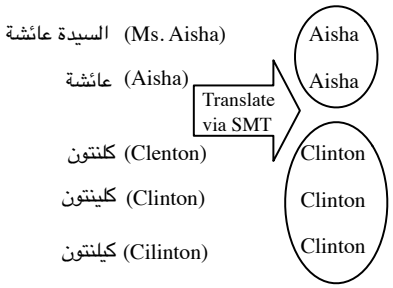


Figure 1: Illustration of translingual projection method for resolving Arabic named entity strings via English space. The English strings in parentheses indicate the literal glosses of the Arabic strings prior to translation.

2.3 Entity context similarity

The context of mentions can play an important role in merging or splitting potential coreferent men-

[1]For example: (Sean Michael Waltman, Sean Waltman) are high confidence-matches even though they are not an exact-string match.

tions. We hypothesize that two mentions in two different documents have a good chance of referring to the same entity if they are mentioned in contexts that are topically very similar. A way of representing a mention context is to consider the words in the mention's neighborhood. The context of a mention can be defined as the words that surround the mention in a window of *n* (50 in our experiments) tokens centered by the mention. In our experiments, we used highly similar contexts to link mentions that might be coreferent.

Computing context similarity between every pair of large number of mentions requires a highly scalable and efficient mechanism. This can be achieved using MapReduce, a distributed computing framework (Dean, 2004)

Elsayed et al. (Elsayed, 2008) proposed an efficient MapReduce solution for the problem of computing the pairwise similarity matrix in large collections. They considered a "bag-of-words" model where similarity of two documents d_i and d_j is measured as follows: $sim(d_i, d_j) = \sum_{t \in d_i \cap d_j} w_{t,d_i} \cdot w_{t,d_j}$, where $w(t, d)$ is the weight of term t in document d. A term contributes to *each* pair that contains it. The list of documents that contain a term is what is contained in the postings of an inverted index. Thus, by processing all postings, the pairwise similarity matrix can be computed by summing term contributions. We use the MapReduce framework for two jobs, inverted indexing and pairwise similarity.

Elsayed et al. suggested an efficient df-cut strategy that eliminates terms that appear in many documents (having high df) and thus contribute less in similarity but cost in computation (e.g., a 99% df-cut means that the most frequent 1% of the terms were discarded). We adopted that approach for computing similarities between the contexts of two mentions. The processing unit was represented as a bag of n words in a window surrounding each mention of a within-document entity. Given a relatively small mention context, we used a high df-cut value of 99.9%.

3 Experiments

We performed our experiments in the context of the Automatic Content Extraction (ACE) evaluation of 2008, run by the National Institute of Standards and Technology (NIST). The evaluation corpus contained approximately 10,000 documents from the following domains: broadcast conversation transcripts, broadcast news transcripts, conversational telephone speech transcripts, newswire, Usenet Newsgroup/Discussion Groups, and weblogs. Systems were required to process the large source sets completely. For performance measurement after the evaluation, NIST selected 412 of the Arabic source documents out of the larger set (NIST, 2008).

For development purposes we used the NIST ACE 2005 Arabic data with within-document ground truth. This consisted of 1,245 documents. We also used exactly 12,000 randomly selected documents from the LDC Arabic Gigaword Third Edition corpus, processed through Serif. The Arabic Gigaword corpus was used to select a threshold of 0.4956 for the context similarity technique via inspection of (A, B) link scores by a native speaker of Arabic.

It must be emphasized that there was no ground truth available for this task in Arabic. Performing this task in the absence of significant training or evaluation data is one emphasis of this work.

3.1 Evaluation measures

We used NIST's scoring techniques to evaluate the performance of our systems. Scoring for the ACE evaluation is done using an scoring script provided by NIST which produces many kinds of statistics. NIST mainly uses a measure called the ACE value, but it also computes B-cubed.

B-Cubed represents the task of finding cross-document entities in the following way: if a user of the system is searching for a particular Bush and finds document D, he or she should be able to find all of the other documents with the same Bush in them as links from D—that is, cross-document entities represent graphs connecting documents. Bagga and Baldwin are able to define precision, recall, and F-measure over a collection of documents in this way.

The ACE Value represents a score similar to B-Cubed, except that every mention and within-document entity is weighted in NIST's specification by a number of factors. Every entity is worth 1 point, a missing entity worth 0, and attribute errors are discounted by multiplying by a factor (0.75 for *CLASS*, 0.5 for *TYPE*, and 0.9 for *SUBTYPE*).

Before scoring can be accomplished, the entities found by the system must be mapped onto those found in the reference provided by NIST. The ACE scorer does this document-by-document,

selecting the mapping that produces the highest score. A description of the evaluation method and entity categorization is available at (NIST, 2008).

3.2 Results and discussion

The results of running the ACE evaluation script on the system output are shown in table 1. The translingual projection system achieves higher scores than all other systems on all measures. Although it achieves only a 2 point improvement over the baseline ACE value, it should be noted that this represents a substantial number of attributes per cross-document entity that it is getting right.

System	Threshold	B-Cubed			ACE Val.
		Prec	Rec	F	
Baseline		37.5	44.1	40.6	19.2
TrnsProj		38.4	44.8	41.3	21.2
CtxtSim	0.2	37.6	35.2	36.4	15.9
CtxtSim	0.3	37.4	43.8	40.3	18.9
CtxtSim	0.4	37.5	44.1	40.6	19.3
CtxtSim	0.4956	37.5	44.1	40.6	19.3
CtxtSim	0.6	37.5	44.1	40.6	19.2

Table 1: Scores from ACE evaluation script.

On the other hand, as the context similarity threshold increases, we notice that the B-Cubed measures reach identical values with the baseline but never exceed it. But as it decreases, it loses B-Cubed recall and ACE value.

While two within-document entities whose longest mention strings match exactly and are legitimately coreferent are likely to be mentioned in the same contexts, it seems that a lower (more liberal) threshold introduces spurious links and creates a different entity clustering.

Translingual projection appears to include links that exact string matching in Arabic does not—part of its purpose is to add close matches to those found by exact string matching. It is able to include these links partly because it allows access to resources in English that are not available for Arabic such as Wikipedia alias lists.

4 Conclusions and Future Work

We have evaluated and discussed a set of techniques for cross-document coreference in Arabic that can be applied in the absence of significant training and evaluation data. As it turns out, an approach based on machine translation is slightly better than a string-matching baseline, across all measures. It worked by using translations from Arabic to English in order to liberalize the string-matching criterion, suggesting that using further techniques via English to discover links may be a fruitful future research path. This also seems to suggest that a Bagga and Baldwin-style vector-space model may not be the first approach to pursue in future work on Arabic.

However, varying other parameters in the context similarity approach should be tried in order to gain a fuller picture of performance. One of them is the df-cut of the MapReduce-based similarity computation. Another is the width of the word token window we used—we may have used one that is too tight to be better than exact Arabic string-matching.

References

Javier Artiles and Satoshi Sekine and Julio Gonzalo 2008. Web People Search—Results of the first evaluation and the plan for the second. *WWW 2008*.

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. *COLING-ACL 1998*.

Y. Chen and J. Martin. 2007. Towards robust unsupervised personal name disambiguation. *EMNLP*.

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics, 33(2)*.

J. Dean and S. Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. *OSDI*.

T. Elsayed and J. Lin and D. W. Oard. 2008. Pairwise Document Similarity in Large Collections with MapReduce. *ACL/HLT*.

Z. Li and S. Khudanpur. 2008. A Scalable Decoder for Parsing-based Machine Translation with Equivalent Language Model State Maintenance. *ACL SSST*.

L. Lloyd and Andrew Mehler and Steven Skiena. 2006. Identifying Co-referential Names Across Large Corpora. *Combinatorial Pattern Matching*.

NIST. 2008. Automatic Content Extraction 2008 Evaluation Plan (ACE08).

The Impact of Query Refinement in the Web People Search Task

Javier Artiles
UNED NLP & IR group
Madrid, Spain
javart@bec.uned.es

Julio Gonzalo
UNED NLP & IR group
Madrid, Spain
julio@lsi.uned.es

Enrique Amigó
UNED NLP & IR group
Madrid, Spain
enrique@lsi.uned.es

Abstract

Searching for a person name in a Web Search Engine usually leads to a number of web pages that refer to several people sharing the same name. In this paper we study whether it is reasonable to assume that pages about the desired person can be filtered by the user by adding query terms. Our results indicate that, although in most occasions there is a query refinement that gives all and only those pages related to an individual, it is unlikely that the user is able to find this expression a priori.

1 Introduction

The Web has now become an essential resource to obtain information about individuals but, at the same time, its growth has made web people search (WePS) a challenging task, because every single name is usually shared by many different people. One of the mainstream approaches to solve this problem is designing meta-search engines that cluster search results, producing one cluster per person which contains all documents referring to this person.

Up to now, two evaluation campaigns – WePS 1 in 2007 (Artiles et al., 2007) and WePS 2 in 2009 (Artiles et al., 2009) – have produced datasets for this clustering task, with over 15 research groups submitting results in each campaign. Since the release of the first datasets, this task is becoming an increasingly popular research topic among Information Retrieval and Natural Language Processing researchers.

For precision oriented queries (for instance, finding the homepage, the email or the phone number of a given person), clustered results might help locating the desired data faster while avoiding confusion with other people sharing the same name. But the utility of clustering is more obvious for recall oriented queries, where the goal is to mine the web for information about a person. In a typical hiring process, for instance, candidates are evaluated not only according to their cv, but also according to their *web profile*, i.e. information about them available in the Web.

One question that naturally arises is whether search results clustering can effectively help users for this task. Eventually, a query refinement made by the user – for instance, adding an affiliation or a location – might have the desired disambiguation effect without compromising recall. The hypothesis underlying most research on Web People Search is that query refinement is risky, because it can enhance precision but it will usually harm recall. Adding the current affiliation of a person, for instance, might make information about previous jobs disappear from search results.

This hypothesis has not, up to now, been empirically confirmed, and it is the goal of this paper. We want to evaluate the actual impact of using query refinements in the Web People Search (WePS) clustering task (as defined in the framework of the WePS evaluation). For this, we have studied to what extent a query refinement can successfully filter relevant results and which type of refinements are the most successful. In our experiments we have considered the search results associated to one individual as a set of relevant documents, and we have tested the ability of different query refinement strategies to retrieve those documents. Our results are conclusive: in most occasions there is a "near-perfect" refinement that filters out most relevant information about a given person, but this refinement is very hard to predict from a user's perspective.

In Section 2 we describe the datasets that where used for our experiments. The experimental methodology and results are presented in Section 3. Finally we present our conclusions in 4.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 361–364,
Suntec, Singapore, 4 August 2009.

2 Dataset

2.1 The WePS-2 corpus

For our experiments we have used the WePS-2 testbed (Artiles et al., 2009) [1]. It consists of 30 datasets, each one related to one ambiguous name: 10 names were sampled from the US Census, 10 from Wikipedia, and 10 from the Computer Science domain (Programme Committee members of the ACL 2008 Conference). Each dataset consists of, at most, 100 web pages written in English and retrieved as the top search results of a web search engine, using the (quoted) person name as query[2].

Annotators were asked to organize the web pages from each dataset in groups where all documents refer to the same person. For instance, the "James Patterson" web results were gruped in four clusters according to the four individuals mentioned with that name in the documents. In cases where a web page refers to more than one person using the same ambiguous name (e.g. a web page with search results from Amazon), the document is assigned to as many groups as necessary. Documents were discarded when there wasn't enough information to cluster them correctly.

2.2 Query refinement candidates

In order to generate query refinement candidates, we extracted several types of features from each document. First, we applied a simple preprocessing to the HTML documents in the corpus, converting them to plain text and tokenizing. Then, we extracted tokens and word n-grams for each document (up to four words lenght). A list of English stopwords was used to remove tokens and n-grams beginning or ending with a stopword. Using the Stanford Named Entity Recognition Tool[3] we obtained the lists of persons, locations and organizations mentioned in each document.

Additionally, we used attributes manually annotated for the WePS-2 Attribute Extraction Task (Sekine and Artiles, 2009). These are person attributes (affiliation, occupation, variations of name, date of birth, etc.) for each individual sharing the name searched. These attributes emulate the kind of query refinements that a user might try in a typical people search scenario.

[1] http://nlp.uned.es/weps

[2] We used the Yahoo! search service API.

[3] http://nlp.stanford.edu/software/CRF-NER.shtml

field	F	prec.	recall	cover.
ae_affiliation	0.99	0.98	1.00	0.46
ae_award	1.00	1.00	1.00	0.04
ae_birthplace	1.00	1.00	1.00	0.09
ae_degree	0.85	0.80	1.00	0.10
ae_email	1.00	1.00	1.00	0.11
ae_fax	1.00	1.00	1.00	0.06
ae_location	0.99	0.99	1.00	0.27
ae_major	1.00	1.00	1.00	0.07
ae_mentor	1.00	1.00	1.00	0.03
ae_nationality	1.00	1.00	1.00	0.01
ae_occupation	0.95	0.93	1.00	0.48
ae_phone	0.99	0.99	1.00	0.13
ae_relatives	0.99	0.98	1.00	0.15
ae_school	0.99	0.99	1.00	0.15
ae_work	0.96	0.95	1.00	0.07
stf_location	0.96	0.95	1.00	0.93
stf_organization	1.00	1.00	1.00	0.98
stf_person	0.98	0.97	1.00	0.82
tokens	1.00	1.00	1.00	1.00
bigrams	1.00	1.00	1.00	0.98
trigrams	1.00	1.00	1.00	1.00
fourgrams	1.00	1.00	1.00	0.98
fivegrams	1.00	1.00	1.00	0.98

Table 1: Results for clusters of size 1

field	F	prec.	recall	cover.
ae_affiliation	0.76	0.99	0.65	0.40
ae_award	0.67	1.00	0.50	0.02
ae_birthplace	0.67	1.00	0.50	0.10
ae_degree	0.63	0.87	0.54	0.15
ae_email	0.74	1.00	0.60	0.16
ae_fax	0.67	1.00	0.50	0.09
ae_location	0.77	1.00	0.66	0.32
ae_major	0.71	1.00	0.56	0.09
ae_mentor	0.75	1.00	0.63	0.04
ae_nationality	0.67	1.00	0.50	0.01
ae_occupation	0.76	0.98	0.65	0.52
ae_phone	0.75	1.00	0.63	0.13
ae_relatives	0.78	0.96	0.68	0.15
ae_school	0.68	0.96	0.56	0.17
ae_work	0.81	1.00	0.72	0.17
stf_location	0.83	0.97	0.77	0.98
stf_organization	0.89	1.00	0.83	1.00
stf_person	0.83	0.99	0.74	0.98
tokens	0.96	0.99	0.94	1.00
bigrams	0.95	1.00	0.92	1.00
trigrams	0.94	1.00	0.92	1.00
fourgrams	0.91	1.00	0.86	0.99
fivegrams	0.89	1.00	0.84	0.99

Table 2: Results for clusters of size 2

field	F	prec.	recall	cover.
ae_affiliation	0.51	0.96	0.39	0.81
ae_award	0.26	1.00	0.16	0.20
ae_birthplace	0.33	0.99	0.24	0.28
ae_degree	0.37	0.90	0.26	0.36
ae_email	0.35	0.96	0.23	0.33
ae_fax	0.30	1.00	0.19	0.15
ae_location	0.34	0.96	0.23	0.64
ae_major	0.30	0.97	0.20	0.22
ae_mentor	0.23	0.95	0.15	0.22
ae_nationality	0.36	0.88	0.26	0.16
ae_occupation	0.52	0.93	0.40	0.80
ae_phone	0.34	0.96	0.23	0.33
ae_relatives	0.32	0.95	0.22	0.16
ae_school	0.40	0.95	0.29	0.43
ae_work	0.45	0.94	0.34	0.38
stf_location	0.62	0.87	0.53	1.00
stf_organization	0.67	0.96	0.56	1.00
stf_person	0.59	0.95	0.47	1.00
tokens	0.87	0.90	0.86	1.00
bigrams	0.79	0.95	0.70	1.00
trigrams	0.75	0.96	0.65	1.00
fourgrams	0.67	0.97	0.55	1.00
fivegrams	0.62	0.96	0.50	1.00

Table 3: Results for clusters of size >=3

3 Experiments

In our experiments we consider each set of documents (cluster) related to one individual in the WePS corpus as a set of relevant documents for a person search. For instance the James Patter-

field	F	prec.	recall	cover.
best-ae	1.00	0.99	1.00	**0.74**
best-all	1.00	1.00	1.00	1.00
best-ner	1.00	1.00	1.00	0.99
best-nl	1.00	1.00	1.00	1.00

Table 4: Results for clusters of size 1

field	F	prec.	recall	cover.
best-ae	0.77	1.00	0.65	**0.79**
best-all	0.95	1.00	0.93	1.00
best-ner	0.92	0.99	0.88	1.00
best-nl	0.96	1.00	0.94	1.00

Table 5: Results for clusters of size 2

field	F	prec.	recall	cover.
best-ae	0.60	0.97	0.47	**0.92**
best-all	0.89	0.96	0.85	1.00
best-ner	0.74	0.95	0.63	1.00
best-nl	0.89	0.95	0.85	1.00

Table 6: Results for clusters of size >=3

son dataset in the WePS corpus contains a total of 100 documents, and 10 of them belong to a British politician named James Patterson. The WePS-2 corpus contains a total of 552 clusters that were used to evaluate the different types of QRs.

For each person cluster, our goal is to find the best query refinements; in an ideal case, an expression that is present in all documents in the cluster, and not present in documents outside the cluster. For each QR type (affiliation, e-mail, n-grams of various sizes, etc.) we consider all candidates found in at least one document from the cluster, and pick up the one that leads to the best harmonic mean ($F_{\alpha=.5}$) of precision and recall on the cluster documents (there might be more than one).

For instance, when we evaluate a set of *token* QR candidates for the politician in the James Patterson dataset we find that among all the tokens that appear in the documents of its cluster, "republican" gives us a perfect score, while "politician" obtains a low precision (we retrieve documents of other politicians named James Patterson).

In some cases a cluster might not have any candidate for a particular type of QR. For instance, manual person attributes like phone number are sparse and won't be available for every individual, whereas tokens and ngrams are always present. We exclude those cases when computing F, and instead we report a *coverage* measure which represents the number of clusters which have at least one candidate of this type of QR. This way we know how often we can use an attribute (coverage) and how useful it is when available (F measure).

field	1	2	>=3
ae_affiliation	20.96	17.88	29.41
ae_occupation	20.25	21.79	24.60
ae_work	3.23	8.38	8.56
ae_location	12.66	12.29	8.02
ae_school	7.03	6.70	6.42
ae_degree	3.23	3.91	5.35
ae_email	5.34	6.15	4.28
ae_phone	6.19	5.03	3.21
ae_nationality	0.28	0.00	3.21
ae_relatives	7.03	5.03	2.67
ae_birthplace	4.22	5.03	1.60
ae_fax	2.95	1.68	1.60
ae_major	3.52	3.91	1.07
ae_mentor	1.41	2.23	0.00
ae_award	1.69	0.00	0.00

Table 7: Distribution of the person attributes used for the "best-ae" strategy

These figures represent a ceiling for each type of query refinement: they represent the efficiency of the query when the user selects the best possible refinement for a given QR type.

We have split the results in three groups depending on the size of the target cluster: (i) rare people, mentioned in only one document (335 clusters of size 1); (ii)people that appear in two documents (92 clusters of size 2), often these documents belong to the same domain, or are very similar; and (iii) all other cases (125 clusters of size >=3).

We also report on the aggregated results for certain subsets of QR types. For instance, if we want to know what results will get a user that picks the best person attribute, we consider all types of attributes (e-mail, affiliation, etc.) for every cluster, and pick up the ones that lead to the best results.

We consider four groups: (i) *best-all* selects the best QR among all the available QR types (ii) *best-ae* considers all manually annotated attributes (iii) *best-ner* considers automatically annotated NEs; and (iv) *best-ng* uses only tokens and ngrams.

3.1 Results

The results of the evaluation for each cluster size (one, two, more than two) are presented in Tables 1, 2 and 3. These tables display results for each QR type. Then Tables 4, 5 and 6 show the results for aggregated QR types.

Two main results can be highlighted: (i) The best overall refinement is, in average, very good ($F = .89$ for clusters of size ≥ 3). In other words, there is usually at least one QR that leads to (approximately) the desired set of results; (ii) this best

refinement, however, is not necessarily an intuitive choice for the user. One would expect users to refine the query with a person's attribute, such as his affiliation or location. But the results for the best (manually extracted) attribute are significantly worse ($F = .60$ for clusters of size ≥ 3), and they cannot always be used (coverage is .74, .79 and .92 for clusters of size 1, 2 and ≥ 3).

The manually tagged attributes from WePS-2 are very precise, although their individual coverage over the different person clusters is generally low. Affiliation and occupation, which are the most frequent, obtain the largest coverage (0.81 and 0.80 for sizes ≥ 3). Also the recall of this type of QRs is low in clusters of two, three or more documents. When evaluating the "best-ae" strategy we found that in many clusters there is at least one manual attribute that can be used as QR with high precision. This is the case mostly for clusters of three or more documents (0.92 coverage) and it decreases with smaller clusters, probably because there is less information about the person and thus less biographical attributes are to be found.

In Table 7 we show the distribution of the actual QR types selected by the "best-ae" strategy. The best type is affiliation, which is selected in 29% of the cases. Affiliation and occupation together cover around half of the cases (54%), and the rest is a long tail where each attribute makes a small contribution to the total. Again, this is a strong indication that the best refinement is probably very difficult to predict a priori for the user.

Automatically recognized named entities in the documents obtain better results, in general, than manually tagged attributes. This is probably due to the fact that they can capture all kinds of related entities, or simply entities that happen to coocur with the person name. For instance, the pages of a university professor that is usually mentioned together with his PhD students could be refined with any of their names. This goes to show that a good QR can be any information related to the person, and that we might need to know the person very well in advance in order to choose this QR.

Tokens and ngrams give us a kind of "upper boundary" of what is possible to achieve using QRs. They include almost anything that is found in the manual attributes and the named entities. They also frequently include QRs that are not realistic for a human refinement. For instance, in clusters of only two documents it is not uncommon that both pages belong to the same domain or that they are near duplicates. In those cases tokens and ngram QR will probably include non informative strings. In some cases the QRs found are neither directly biographical or related NEs, but topical information (e.g. the term "soccer" in the pages of a football player or the ngram "alignment via structured multilabel" that is the title of a paper written by a Computer Science researcher). These cases widen even more the range of effective QRs. The overall results of using tokens and ngrams are almost perfect for all clusters, but at the cost of considering every possible bit of information about the person or even unrelated text.

4 Conclusions

In this paper we have studied the potential effects of using query refinements to perform the Web People Search task. We have shown that although in theory there are query refinements that perform well to retrieve the documents of most individuals, the nature of these ideal refinements varies widely in the studied dataset, and there is no single intuitive strategy leading to robust results. Even if the attributes of the person are well known beforehand (which is hardly realistic, given that in most cases this is precisely the information needed by the user), there is no way of anticipating which expression will lead to good results for a particular person. These results confirm that search results clustering might indeed be of practical help for users in Web people search.

References

Javier Artiles, Julio Gonzalo, and Satoshi Sekine. 2007. The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. ACL.

Javier Artiles, Julio Gonzalo, and Satoshi Sekine. 2009. Weps 2 evaluation campaign: overview of the web people search clustering task. In *WePS 2 Evaluation Workshop. WWW Conference 2009*.

Satoshi Sekine and Javier Artiles. 2009. Weps2 attribute extraction task. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*.

Composite Kernels For Relation Extraction

Frank Reichartz
Fraunhofer IAIS
St. Augustin, Germany

Hannes Korte
Fraunhofer IAIS
St. Augustin, Germany

Gerhard Paass
Fraunhofer IAIS
St. Augustin, Germany

{frank.reichartz,hannes.korte,gerhard.paass}@iais.fraunhofer.de

Abstract

The automatic extraction of relations between entities expressed in natural language text is an important problem for IR and text understanding. In this paper we show how different kernels for parse trees can be combined to improve the relation extraction quality. On a public benchmark dataset the combination of a kernel for phrase grammar parse trees and for dependency parse trees outperforms all known tree kernel approaches alone suggesting that both types of trees contain complementary information for relation extraction.

1 Introduction

The same semantic relation between entities in natural text can be expressed in many ways, e.g. "Obama was educated at Harvard", "Obama is a graduate of Harvard Law School", or, "Obama went to Harvard College". Relation extraction aims at identifying such semantic relations in an automatic fashion.

As a preprocessing step named entity taggers detect persons, locations, schools, etc. mentioned in the text. These techniques have reached a sufficient performance level on many datasets (Tjong et al., 2003). In the next step relations between recognized entities, e.g. person-educated-in-school(Obama,Harvard) are identified.

Parse trees provide extensive information on syntactic structure. While feature-based methods may compare only a limited number of structural details, kernel-based methods may explore an often exponential number of characteristics of trees without explicitly representing the features. Zelenko et al. (2003) and Culotta and Sorensen (2004) proposed kernels for dependency trees (DTs) inspired by string kernels. Zhang et al. (2006) suggested a kernel for phrase grammar parse trees. Bunescu and Mooney (2005) investigated a kernel that computes similarities between nodes on the shortest path of a DT connecting the entities. Reichartz et al. (2009) presented DT kernels comparing substructures in a more sophisticated way.

Up to now no studies exist on how kernels for different types of parse trees may support each other. To tackle this we present a study on how those kernels for relation extractions can be combined. We implement four state-of-the-art kernels. Subsequently we combine pairs of kernels linearly or by polynomial expansion. On a public benchmark dataset we show that the combined phrase grammar parse tree kernel and dependency parse tree kernel outperforms all others by 5.7% F-Measure reaching an F-Measure of 71.2%. This result shows that both types of parse trees contain relevant information for relation extraction.

The remainder of the paper is organized as follows. In the next section we describe the investigated tree kernels. Subsequently we present the method to combine two kernels. The fourth section details the experiments on a public benchmark dataset. We close with a summary and conclusions.

2 Kernels for Relation Extraction

Relation extraction aims at learning a relation from a number of positive and negative instances in natural language sentences. As a classifier we use Support Vector Machines (SVMs) (Joachims, 1999) which can compare complex structures, e.g. trees, by kernels. Given the kernel function, the SVM tries to find a hyperplane that separates positive from negative examples of the relation. This type of max-margin separator has been shown both empirically and theoretically to provide good generalization performance on new examples.

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 365–368,
Suntec, Singapore, 4 August 2009.

2.1 Parse Trees

A sentence can be processed by a parser to generate a parse tree, which can be further categorized in phrase grammar parse trees (PTs) and dependency parse trees (DTs). For DTs there is a bijective mapping between the words in a sentence and the nodes in the tree. DTs have a natural ordering of the children of the nodes induced by the position of the corresponding words in the sentence. In contrast PTs introduce new intermediate nodes to better express the syntactical structures of a sentence in terms of phrases.

2.2 Path-enclosed PT Kernel

The *Path-enclosed PT Tree Kernel* (Zhang et al., 2008) operates on PTs. It is based on the *Convolution Tree Kernel* of Collins and Duffy (2001). The *Path-enclosed Tree* is the parse tree pruned to the nodes that are connected to leaves (words) that belong to the path connecting both relation entities. The leaves (and connected inner nodes) in front of the first relation entity node and behind the second one are simply removed. In addition, for the entities there are new artificial nodes labeled with the relation argument index, and the entity type. Let $K_{\text{CD}}(T_1, T_2)$ be the *Convolution Tree Kernel* (Collins and Duffy, 2001) of two trees T_1, T_2, then the *Path-enclosed PT Kernel* (ZhangPT) is defined as

$$K_{\text{ZhangPT}}(X, Y) = K_{\text{CD}}(X^*, Y^*)$$

where X^* and Y^* are the subtrees of the original tree pruned to the nodes enclosed by the path connecting the two entities in the phrase grammer parse trees as described by Zhang et al. (2008).

2.3 Dependency Tree Kernel

The *Dependency Tree Kernel* (DTK) of Culotta and Sorensen(2004) is based on the work of Zelenko et al. (2003). It employs a *node kernel* $\Delta(u, v)$ measuring the similarity of two tree nodes u, v and its substructures. Nodes may be described by different features like POS-tags, chunk tags, etc.. If the corresponding word describes an entity, the entity type and the mention is provided. To compare relations in two instance sentences X, Y Culotta and Sorensen (2004) proposes to compare the subtrees induced by the relation arguments x_1, x_2 and y_1, y_2, i.e. computing the node kernel between the two lowest common ancestors (lca) in the dependecy tree of the relation argument nodes

$$K_{\text{DTK}}(X, Y) = \Delta(lca(x_1, x_2), lca(y_1, y_2))$$

The node kernel $\Delta(u, v)$ is definend over two nodes u and v as the sum of the node similarity and their children similarity. The *children similarity function* $C(s, t)$ uses a modified version of the *String Subsequence Kernel* of Shawe-Taylor and Christianini (2004) to compute recursively the sum of node kernel values of subsequences of node sequences s and t. The function $C(s, t)$ sums up the similarities of all subsequences in which every node matches its corresponding node.

2.4 All-Pairs Dependency Tree Kernel

The All-Pairs Dependency Tree Kernel (All-Pairs-DTK) (Reichartz et al., 2009) sums up the node kernels of all possible combinations of nodes contained in the two subtrees implied by the relation argument nodes as

$$K_{\text{All-Pairs}}(X, Y) = \sum_{u \in V_x} \sum_{v \in V_y} \Delta(u, v)$$

where V_x and V_y are sets containing the nodes of the complete subtrees rooted at the respective lowest common ancestors. The consideration of all possible pairs of nodes and their similarity ensure that relevant information in the subtrees is utilized.

2.5 Dependency Path Tree Kernel

The Dependency Path Tree Kernel (Path-DTK) (Reichartz et al., 2009) not only measures the similarity of the root nodes and its descendents (Culotta and Sorensen, 2004) or the similarities of nodes on the path (Bunescu and Mooney, 2005). It considers the similarities of all nodes (and substructures) using the node kernel Δ on the path connecting the two relation argument entity nodes. To this end the pairwise comparison is performed using the ideas of the subsequence kernel of Shawe-Taylor and Cristianini (2004), therefore relaxing the "same length" restriction of (Bunescu and Mooney, 2005). The Path-DTK effectively compares the nodes from paths with different lengths while maintaining the ordering information and considering the similarities of substructures.

The parameter q is the upper bound on the node distance whereas the parameter μ, $0 < \mu \leq 1$, is a factor that penalizes gaps. The Path-DTK is

Kernel	5-times 5-fold Cross-Validation on Training Set						Test Set					
	At	Part	Role	Prec	Rec	F	At	Part	Role	Prec	Rec	F
DTK	54.9	52.8	72.3	71.7	53.7	61.4 (0.32)	50.3	43.4	68.5	79.5	44.0	56.7
All-Pairs-DTK	59.1	53.6	73.0	73.1	57.8	64.5 (0.26)	54.3	53.9	71.8	80.2	49.6	61.3
Path-DTK	64.8	62.9	77.2	80.2	61.2	69.4 (0.09)	54.9	55.6	73.5	76.7	52.8	62.5
ZhangPT	66.8	69.1	77.7	80.6	65.0	71.9 (0.21)	62.9	64.2	72.2	82.0	54.5	65.5
ZhangPT + Path-DTK	**70.1**	**76.6**	**80.8**	**84.6**	**68.2**	**75.5** (0.20)	**66.3**	**71.3**	**77.7**	**85.7**	**60.9**	**71.2**

Table 1: F-values for 3 selected relations and micro-averaged precision, recall and F-score (with standard error) for all 5 relations on the training (CV) and test set in percent.

defined as
$K_{\text{Path-DTK}}(X, Y) =$

$$\sum_{\substack{\mathbf{i}\in I_{|x|}, \mathbf{j}\in I_{|y|},\\ |\mathbf{i}|=|\mathbf{j}|,\, d(\mathbf{i}),d(\mathbf{j})\leq q}} \mu^{d(\mathbf{i})+d(\mathbf{j})}\Delta'(x(\mathbf{i}), y(\mathbf{j}))$$

where x and y are the paths in the dependency tree between the relation arguments and $x(\mathbf{i})$ is the subsequence of the nodes indexed by $\mathbf{i}$, analogously for $\mathbf{j}$. I_k is the set of all possible index sequences with highest index k and $d(i) = max(\mathbf{i}) - min(\mathbf{i}) + 1$ is the covered distance. The function Δ' is the sum of the pairwise applications of the node kernel Δ.

3 Kernel composition

In this paper we use the following two approaches to combine two normalized[1] kernels K_1, K_2 (Schoelkopf and Smola, 2001). For a weighting factor α we have the composite kernel:

$$K_c(X, Y) = \alpha K_1(X, Y) + (1 - \alpha)K_2(X, Y)$$

Furthermore it is possible to use polynomial expansion on the single kernels, i.e. $K^p(X, Y) = (K(X, Y) + 1)^p$. Our experiments are performed with $\alpha = 0.5$ and the sum of linear kernels (L) or poly kernels (P) with $p = 2$.

4 Experiments

In this section we present the results of the experiments with kernel-based methods for relation extraction. Throughout this section we will compare the approaches considering their classification quality on the publicly available benchmark dataset ACE-2003 (Mitchell et al., 2003). It consists of news documents containing 176825 words splitted in a test and training set. Entities and the relations between them were manually annotated. The entities are marked by the types *named* (e.g. "Albert Einstein") , *nominal* (e.g. "University") and *pronominal* (e.g. "he"). There are 5 top level relation types *role*, *part*, *near*, *social* and *at*, which are further differentiated into 24 subtypes.

4.1 Experimental Setup

We implemented the tree-kernels for relation extraction in Java and used Joachim's (1999) SVM^{light} with the JNI Kernel Extension using the implementation details from the original papers. For the generation of the parse trees we used the Stanford Parser (Klein and Manning, 2003). We restricted our experiments to relations between named entities, where NER approaches may be used to extract the arguments. Without any modification the kernels could also be applied to the all types setting as well. We conducted classification tests on the five top level relations of the dataset. For each relation we trained a separate SVM following the one vs. all scheme for multi-class classification. We also employed a standard grid-search on the training set with a 5-times repeated 5-fold cross validation to optimize the parameters of all kernels as well as the SVM-parameter C for the classification runs on the separate test set. We use the standard evaluation measures for classification accuracy: precision, recall and F-measure.

4.2 Results

Table 1 shows F-values for three selected relations and micro-averaged results for all 5 relations on the training and test set. In addition the F-scores for the three relations containing the most instances are provided. Kernel and SVM parameters are optimized solely on the training set. Note that the training set results were obtained on the left-out folds of cross-validation. The composite kernel ZhangPT + Path-DTK performs the best on the cross validations run as well as on the test-set. It outperforms all previously suggested solutions

[1]Kernel normalization: $K_n(X, Y) = \frac{K(X,Y)}{\sqrt{K(X,X)\cdot K(Y,Y)}}$

	DTK	All-Pairs-DTK	Path-DTK	ZhangPT
ZhangPT	63.5 (70.2) PP	67.9 (72.8) PP	**71.2 (75.5)** LP	65.5 (71.9)
Path-DTK	62.7 (67.7) PP	62.9 (69.5) PL	62.5 (69.4)	
All-Pairs-DTK	60.0 (64.7) PP	61.3 (64.5)		
DTK	56.7 (61.4)			

Table 2: Micro-averaged F-values for the Single and Combined Kernels on the Test Set (outside parenthesis) and with 5-times repeated 5-fold CV on the Training Set (inside parenthesis). LP denotes the combination type linear and polynomial, analogously PP and PL.

by at least 5.7% F-Measure on the prespecified test-set and by 3.6% F-Measure on the cross validation. Table 2 shows the F-values of the different combinational kernels on the test set as well as on the cross validation on the training set. The ZhangPT + Path-DTK performs the best out of all possible combinations. The difference in F-values between ZhangPT + Path-DTK and ZhangPT is according to corrected resampled t-test (Bouckaert and Frank, 2004) significant at a level of 99.9%. These results show that the simultanous consideration of phrase grammar parse trees and dependency parse trees by the combination of the two kernels is meaningful for relation extraction.

5 Conclusion and Future Work

In this paper we presented a study on the combination of state of the art kernels to improve relation extraction quality. We were able to show that a combination of a kernel for phrase grammar parse trees and one for dependency parse trees outperforms all other published parse tree kernel approaches indicating that both kernels captures complementary information for relation extraction. A promising direction for future work is the usage of more sophisticated features aiming at capturing the semantics of words e.g. word sense disambiguation (Paaß and Reichartz, 2009). Other promising directions are the study on the applicability of the kernel to other languages and exploring combinations of more than two kernels.

6 Acknowledgement

The work presented here was funded by the German Federal Ministry of Economy and Technology (BMWi) under the THESEUS project.

References

Remco R. Bouckaert and Eibe Frank. 2004. Evaluating the replicability of significance tests for comparing learning algorithms. In *PAKDD '04*.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proc. HLT/EMNLP*, pages 724 – 731.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proc. NIPS '01*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *ACL '04*.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL '03*.

Alexis Mitchell et al. 2003. ACE-2 Version 1.0; Corpus LDC2003T11. Linguistic Data Consortium.

Gerhard Paaß and Frank Reichartz. 2009. Exploiting semantic constraints for estimating supersenses with CRFs. In *Proc. SDM 2009*.

Frank Reichartz, Hannes Korte, and Gerhard Paass. 2009. Dependency tree kernels for relation extraction from natural language text. In *ECML '09*.

Bernhard Schoelkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Erik F. Tjong, Kim Sang, and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *CoRR cs.CL/0306050:*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.

Min Zhang, Jie Zhang, and Jian Su. 2006. Exploring syntactic features for relation extraction using a convolution tree kernel. In *Proc. HLT/NAACL'06*.

Min Zhang, GuoDong Zhou, and Aiti Aw. 2008. Exploring syntactic structured features over parse trees for relation extraction using kernel methods. *Inf. Process. Manage.*, 44(2):687–701.

Predicting Unknown Time Arguments based on Cross-Event Propagation

Prashant Gupta
Indian Institute of Information Technology Allahabad
Allahabad, India, 211012
greatprach@gmail.com

Heng Ji
Computer Science Department, Queens College and the Graduate Center, City University of New York
New York, NY, 11367, USA
hengji@cs.qc.cuny.edu

Abstract

Many events in news articles don't include time arguments. This paper describes two methods, one based on rules and the other based on statistical learning, to predict the unknown time argument for an event by the propagation from its related events. The results are promising – the rule based approach was able to correctly predict 74% of the unknown event time arguments with 70% precision.

1 Introduction

Event time argument detection is important to many NLP applications such as textual inference (Baral et al., 2005), multi-document text summarization (e.g. Barzilay e al., 2002), temporal event linking (e.g. Bethard et al., 2007; Chambers et al., 2007; Ji and Chen, 2009) and template based question answering (Ahn et al., 2006). It's a challenging task in particular because about half of the event instances don't include explicit time arguments. Various methods have been exploited to identify or infer the implicit time arguments (e.g. Filatova and Hovy, 2001; Mani et al., 2003; Lapata and Lascarides, 2006; Eidelman, 2008).

Most of the prior work focused on the sentence level by clustering sentences into topics and ordering sentences on a time line. However, many sentences in news articles include multiple events with different time arguments. And it was not clear how the errors of topic clustering techniques affected the inference scheme. Therefore it will be valuable to design inference methods for more fine-grained events.

In addition, in the previous approaches the linguistic evidences such as verb tense were mainly applied for inferring the exact dates of implicit time expressions. In this paper we are interested in those more challenging cases in which an event mention and all of its coreferential event mentions do not include any explicit or implicit time expressions; and therefore its time argument can only be predicted based on other related events even if they have different event types.

2 Terminology and Task

In this paper we will follow the terminology defined in the Automatic Content Extraction (ACE)[1] program:

entity: an object or a set of objects in one of the semantic categories of interest: persons, locations, organizations, facilities, vehicles and weapons.

event: a specific occurrence involving participants. The 2005 ACE evaluation had 8 types of events, with 33 subtypes; for the purpose of this paper, we will treat these simply as 33 distinct event types. In contrast to ACE event extraction, we exclude generic, negative, and hypothetical events.

event mention: a phrase or sentence within which an event is described.

event argument: an entity involved in an event with some specific role.

event time: an exact date normalized from time expressions and a role to indicate that an event occurs before/after/within the date.

For any pair of event mentions <EM_i, EM_j>, if:

- EM_i includes a time argument *time-arg*;
- EM_j and its coreferential event mentions don't include any time arguments;

The goal of our task is to determine whether *time-arg* can be propagated into EM_j or not.

3 Motivation

The events in a news document may contain a temporal or locative dimension, typical about an unfolding situation. Various situations are evolving, updated, repeated and corrected in different event mentions. Here later information may override earlier more tentative or incomplete

[1] http://www.nist.gov/speech/tests/ace/

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 369–372,
Suntec, Singapore, 4 August 2009.

events. As a result, different events with particular types tend to occur together frequently, for example, the chains of "Conflict→Life-Die/Life-Injure" and "Justice-Convict → Justice-Charge-Indict/Justice-Trial-Hearing" often appear within one document. To avoid redundancy, the news writers rarely provide time arguments for all of these events. Therefore, it's possible to recover the time argument of an event by gleaning knowledge from its related events, especially if they are involved in a pre-cursor/consequence or causal relation. We present two examples as follows.

- **Example 1**

For example, we can propagate the time "Sunday (normalized into "2003-04-06")" from a "Conflict-Attack" EM_i to a "Life-Die" EM_j because they both involve "Kurdish/Kurds":

[***Sentence including EM$_i$***]
Injured Russian diplomats and a convoy of America's *Kurdish* comrades in arms were among unintended victims caught in **crossfire** and friendly fire ***Sunday***.
[***Sentence including EM$_j$***]
Kurds said 18 of their own **died** in the mistaken U.S. air **strike**.

- **Example 2**

This kind of propagation can also be applied between two events with similar event types. For example, in the following we can propagate "Saturday" from a "Justice-Convict" event to a "Justice-Sentence" event because they both involve arguments "A state security court/state" and "newspaper/Monitor":

[***Sentence including EM$_i$***]
A state security court suspended a *newspaper* critical of the government **Saturday** after **convicting** it of publishing religiously inflammatory material.
[***Sentence including EM$_j$***]
The **sentence** was the latest in a series of *state* actions against the *Monitor*, the only English language daily in Sudan and a leading critic of conditions in the south of the country, where a civil war has been waged for 20 years.

4 Approaches

Based on these motivations we have developed two approaches to conduct cross-event propagation. Section 4.1 below will describe the rule-based approach and section 4.2 will present the statistical learning framework respectively.

4.1 Rule based Prediction

The easiest solution is to encode rules based on constraints from event arguments and positions of two events. We design three types of rules in this paper.

If EM_i has an event type $type_i$ and includes an argument arg_i with role $role_i$, while EM_j has an event type $type_j$ and includes an argument arg_j with role $role_j$, they are not from two temporally separate groups of Justice events {Release-Parole, Appeal, Execute, Extradite, Acquit, Pardon} and {Arrest-Jail, Trial-Hearing, Charge-Indict, Sue, Convict, Sentence, Fine}[2], and they match one of the following rules, then we propagate the time argument between them.

- **Rule1: Same-Sentence Propagation**

  EM_i and EM_j are in the same sentence and only one time expression exists in the sentence; This follows the within-sentence inference idea in (Lapata and Lascarides, 2006).

- **Rule2: Relevant-Type Propagation**

  arg_i is coreferential with arg_j;
  $type_i$= "Conflict", $type_j$= "Life-Die/Life-Injure";
  $role_i$="Target" and $role_j$="Victim", or
  $role_i$=$role_j$="Instrument".

- **Rule3: Same-Type Propagation**

  arg_i is coreferential with arg_j, $type_i$= $type_j$, $role_i$= $role_j$, and they match one of the *Time-Cue* event type and argument role combinations in Table 1.

Event Type$_i$	Argument Role$_i$
Conflict	Target/Attacker/Crime
Justice	Defendant/Crime/Plantiff
Life-Die/Life-Injure	Victim
Life-Be-Born/Life-Marry/Life-Divorce	Person/Entity
Movement-Transport	Destination/Origin
Transaction	Buyer/Seller/Giver/ Recipient
Contact	Person/Entity
Personnel	Person/Entity
Business	Organization/Entity

Table 1. Time-Cue Event Types and Argument Roles

The combinations shown in Table 1 above are those informative arguments that are specific enough to indicate the event time, thus they are

[2] Statistically there is often a time gap between these two groups of events.

called "Time-Cue" roles. For example, in a "Conflict-Attack" event, "Attacker" and "Target" are more important than "Person" to indicate the event time. The general idea is similar to extracting the cue phrases for text summarization (Edmundson, 1969).

4.2 Statistical Learning based Prediction

In addition, we take a more general statistical approach to capture the cross-event relations and predict unknown time arguments. We manually labeled some ACE data and trained a Maximum Entropy classifier to determine whether to propagate the time argument of EM_i to EM_j or not. The features in this classifier are most derived from the rules in the above section 4.1.

Following Rule 1, we build the following two features:

- **Feature1: Same-Sentence**

  F_SameSentence: whether EM_i and EM_j are located in the same sentence or not.

- **Feature2: Number of Time Arguments**

  F_TimeNum: if *F_*SameSentence = true, then assign the number of time arguments in the sentence, otherwise assign the feature value as "Empty".

For all the Time-Cue argument role pairs in Rule 2 and Rule 3, we construct a set of features:

- **Feature Set3: Time-Cue Argument Role Matching**

  $F\_CueRole_{ij}$: Construct a feature for any pair of Time-Cue role types $Role_i$ and $Role_j$ in Rule 2 and 3, assign the feature value as follows:
  if the argument arg_i in EM_i has a role $Role_i$ and the argument arg_j has a role $Role_j$:
  if arg_i and arg_j are coreferential then $F\_CueRole_{ij}$ = *Coreferential*,
  else $F\_CueRole_{ij}$ = *Non-Coreferential.*
  else $F\_CueRole_{ij}$ = *Empty*.

5 Experimental Results

In this section we present the results of applying these two approaches to predict unknown event time arguments.

5.1 Data and Answer-Key Annotation

We used 47 newswire texts from ACE 2005 training corpora to train the Maximum Entropy classifier, and conduct blind test on a separate set of 10 ACE 2005 newswire texts. For each document we constructed any pair of event mentions $<EM_i, EM_j>$ as a candidate sample if EM_i includes a time argument while EM_j and its coreferential event mentions don't include any time arguments. We then manually labeled "Propagate/Not-Propagate" for each sample. The annotation for both training and test sets took one human annotator about 10 hours. We asked another annotator to label the 10 test texts separately and the inter-annotator agreement is above 95%. There are 485 "Propagate" samples and 617 "Not-Propagate" samples in the training set; and in total 212 samples in the test set.

5.2 Overall Performance

Table 2 presents the overall Precision (P), Recall (R) and F-Measure (F) of using these two different approaches.

Method	P (%)	R (%)	F(%)
Rule-based	70.40	74.06	72.18
Statistical Learning	72.48	50.94	59.83

Table 2. Overall Performance

The results of the rule-based approach are promising: we are able to correctly predict 74% of the unknown event time arguments at about 30% error rate. The most common correctly propagated pairs are:

- From Conflict-Attack to Life-Die/Life-Injure
- From Justice Convict to Justice-Sentence/ Justice-Charge-Indict
- From Movement-Transport to Contact-Meet
- From Justice-Charge-Indict to Justice-Convict

5.3 Discussion

From Table 2 we can see that the rule-based approach achieved 23% higher recall than the statistical classifier, with only 2% lower precision. The reason is that we don't have enough training data to capture all the evidences from different Time-cue roles. For instance, for the Example 2 in section 3, Rule 3 is able to predict the time argument of the "Justice-Sentence" event as "Saturday (normalized as 2003-05-10)" because these two events share the coreferential Time-cue "Defendant" arguments "newspaper" and "Monitor". However, there is only one positive sample matching these conditions in the training corpora, and thus the Maximum Entropy classifier assigned a very low confidence score for propagation. We have also tried to combine these two approaches in a self-training framework – adding the results from the propagation rules as additional training data and re-train the Maximum

Entropy classifier, but it did not provide further improvement.

The spurious errors made by the prediction rules reveal both the shortcomings of ignoring event reporting order and the restricted matching on event arguments.

For example, in the following sentences:

[***Context Sentence***]

American troops stormed a presidential palace and other key buildings in Baghdad as U.S. tanks rumbled into the heart of the battered Iraqi capital on ***Monday*** amid the thunder of **gunfire** and **explosions**…

[***Sentence including EM$_j$***]

At the palace compound, Iraqis **shot** <*instrument*>*small arms*</*instrument*> fire from a clock tower, which the U.S. tanks quickly destroyed.

[***Sentence including EM$_i$***]

The first one was on ***Saturday*** and triggered intense <*instrument*>*gun*</*instrument*> **battles**, which according to some U.S. accounts, left at least 2,000 Iraqi fighters dead.

The time argument "Saturday" was mistakenly propagated from the "Conflict-Attack" event "battles" to "shot" because they share the same Time-cue role "instrument" ("small arms/gun"). However, the correct time argument for the "shot" event should be "Monday" as indicated in the "gunfire/explosions" event in the previous context sentence. But since the "shot" event doesn't share any arguments with "gunfire/explosions", our approach failed to obtain any evidence for propagating "Monday". In the future we plan to incorporate the distance and event reporting order as additional features and constraints.

Nevertheless, as Table 2 indicates, the rewards of using propagation rules outweigh the risks because it can successfully predict a lot of unknown time arguments which were not possible using the traditional time argument extraction techniques.

6 Conclusion and Future Work

In this paper we described two approaches to predict unknown time arguments based on the inference and propagation between related events. In the future we shall improve the confidence estimation of the Maximum Entropy classifier so that we could incorporate dynamic features from the high-confidence time arguments which have already been predicted. We also plan to test the effectiveness of this system in textual inference, temporal event linking and event coreference resolution. We are also interested in extending these approaches to the setting of cross-document, so that we can predict more time arguments based on the background knowledge from related documents.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency under Contract No. HR0011-06-C-0023 via 27-001022, and the CUNY Research Enhancement Program and GRTI Program.

References

David Ahn, Steven Schockaert, Martine De Cock and Etienne Kerre. 2006. Supporting Temporal Question Answering: Strategies for Offline Data Collection. *Proc. 5th International Workshop on Inference in Computational Semantics (ICoS-5).*

Regina Barzilay, Noemie Elhadad and Kathleen McKeown. 2002. Inferring Strategies for Sentence Ordering in Multidocument Summarization. *JAIR, 17:35-55*.

Chitta Baral, Gregory Gelfond, Michael Gelfond and Richard B. Scherl. 2005. *Proc. AAAI'05 Workshop on Inference for Textual Question Answering.*

Steven Bethard, James H. Martin and Sara Klingenstein. 2007. Finding Temporal Structure in Text: Machine Learning of Syntactic Temporal Relations. *International Journal of Semantic Computing (IJSC)*, 1(4), December 2007.

Nathanael Chambers, Shan Wang and Dan Jurafsky. 2007. Classifying Temporal Relations Between Events. *Proc. ACL2007.*

H. P. Edmundson. 1969. New Methods in Automatic Extracting. *Journal of the ACM.* 16(2):264-285.

Vladimir Eidelman. 2008. Inferring Activity Time in News through Event Modeling. *Proc. ACL-HLT 2008.*

Elena Filatova and Eduard Hovy. 2001. Assigning Time-Stamps to Event-Clauses. *Proc. ACL 2001 Workshop on Temporal and Spatial Information Processing.*

Heng Ji and Zheng Chen. 2009. Cross-document Temporal and Spatial Person Tracking System Demonstration. *Proc. HLT-NAACL 2009.*

Mirella Lapata and Alex Lascarides. 2006. Learning Sentence-internal Temporal Relations. *Journal of Artificial Intelligence Research* 27. pp. 85-117.

Inderjeet Mani, Barry Schiffman and Jianping Zhang. 2003. Inferring Temporal Ordering of Events in News. *Proc. HLT-NAACL 2003.*

Author Index