2006

COLING • ACL

COLING • ACL 2006

21st International Conference on
Computational Linguistics
and
44th Annual Meeting of the
Association for Computational Linguistics

Proceedings of the Interactive Presentation Sessions

17-18 July 2006
Sydney, Australia

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

Preface

The COLING/ACL 2006 Interactive Presentations took place on Monday 17th and Tuesday 18th July, 2006 as part of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics held in Sydney, Australia.

The presentations allow developers of *implemented* computational linguistics software systems and libraries the opportunity to describe the design, development and functionality of their work in an interactive setting. It was also an opportunity to gain direct feedback from their users, and exchange ideas and development techniques with other developers.

The presentations are the next iteration of the ongoing evolution of Demonstration and/or Interactive Poster sessions held at previous ACL annual meetings. Traditionally, the demonstrations have had an emphasis on mature systems and practical applications. Last year, Masaaki Nagata and Ted Pederson encouraged the dissemination of novel ideas supported by an implementation with the introduction of *Interactive Posters*.

This year continued the emphasis on the interactive nature of this forum for developers of systems and libraries. The presentations were a combination of short conference-like talks and interactive demonstrations with audience involvement, questions and comments strongly encouraged. The session title *Interactive Presentations* was a challenge to the presenters – to fully exploit the opportunity to pro-actively engage more closely with the audience.

There were 31 proposals for interactive presentations submitted and 20 were accepted (an acceptance rate of 64.5%) after full peer review of the 4-page descriptions included in this volume and an additional 2-page presentation script. As well as the usual scholarship and technical criteria, the reviews took into consideration whether the software was available and ready to use, and the degree of interactivity proposed in the presentation script.

I would like to thank the general conference chair, Nicoletta Calzolari, the main program chairs, Claire Cardie and Pierre Isabelle, and especially the local organisers, Robert Dale and Cécile Paris, for their patience, advice and encouragement while I made many mistakes during my first involvement with conference organising. Thanks also to Tim Baldwin, Olivia Kwong and Menno van Zaanen, for their patience and help with the presentation announcements, compiling the proceedings and keeping track of the deadlines. Finally, thanks to Judy Potter and her team for handling the myriad of local space, time and audio-visual arrangements.

James Curran
University of Sydney

Interactive Presentations chair

Interactive Presentations Program

Monday, 17 July 2006

10:30–12:00 **Session IP1A: Evaluation Systems**

FAST – An Automatic Generation System for Grammar Tests
Chia-Yin Chen, Hsien-Chin Liou and Jason S. Chang

Is It Correct? – Towards Web-Based Evaluation of Automatic Natural Language Phrase Generation
Calkin S. Montero and Kenji Araki

10:30–12:00 **Session IP1B: Resource Development Systems**

LeXFlow: A System for Cross-Fertilization of Computational Lexicons
Maurizio Tesconi, Andrea Marchetti, Francesca Bertagna, Monica Monachini, Claudia Soria and Nicoletta Calzolari

Valido: A Visual Tool for Validating Sense Annotations
Roberto Navigli

15:00–16:30 **Session IP2A: Information Retrieval Systems**

An Intelligent Search Engine and GUI-based Efficient MEDLINE Search Tool Based on Deep Syntactic Parsing
Tomoko Ohta, Yusuke Miyao, Takashi Ninomiya, Yoshimasa Tsuruoka, Akane Yakushiji, Katsuya Masuda, Jumpei Takeuchi, Kazuhiro Yoshida, Tadayoshi Hara, Jin-Dong Kim, Yuka Tateisi and Jun'ichi Tsujii

MIMA Search: A Structuring Knowledge System towards Innovation for Engineering Education
Hideki Mima

15:00–16:30 **Session IP2B: Question Answering Systems**

FERRET: Interactive Question-Answering for Real-World Environments
Andrew Hickl, Patrick Wang, John Lehmann and Sanda Harabagiu

K-QARD: A Practical Korean Question Answering Framework for Restricted Domain
Young-In Song, HooJung Chung, Kyoung-Soo Han, JooYoung Lee, Hae-Chang Rim and Jae-Won Lee

Tuesday, 18 July 2006

10:30–12:00 **Session IP3A: Techniques and Tools**

An Intermediate Representation for the Interpretation of Temporal Expressions
Paweł Mazur and Robert Dale

Chinese Named Entity and Relation Identification System
Tianfang Yao and Hans Uszkoreit

10:30–12:00 **Session IP3B: Techniques and Systems**

Computational Analysis of Move Structures in Academic Abstracts
Jien-Chen Wu, Yu-Chia Chang, Hsien-Chin Liou and Jason S. Chang

LexNet: A Graphical Environment for Graph-Based NLP
Dragomir R. Radev, Güneş Erkan, Anthony Fader, Patrick Jordan, Siwei Shen and James P. Sweeney

13:15–14:45 **Session IP4A: Machine Translation Systems**

Archivus: A Multimodal System for Multimedia Meeting Browsing and Retrieval
Marita Ailomaa, Miroslav Melichar, Agnes Lisowska, Martin Rajman and Susan Armstrong

Re-Usable Tools for Precision Machine Translation
Jan Tore Lønning and Stephan Oepen

13:15–14:45 **Session IP4B: Multimodal Systems**

The SAMMIE System: Multimodal In-Car Dialogue
Tilman Becker, Peter Poller, Jan Schehl, Nate Blaylock, Ciprian Gerstenberger and Ivana Kruijff-Korbayová

TwicPen: Hand-held Scanner and Translation Software for non-Native Readers
Eric Wehrli

Tuesday, 18 July 2006 (continued)

15:00–16:30 **Session IP5A: Tools and Systems**

An Implemented Description of Japanese: The Lexeed Dictionary and the Hinoki Treebank
Sanae Fujita, Takaaki Tanaka, Francis Bond and Hiromi Nakaiwa

NLTK: The Natural Language Toolkit
Steven Bird

15:00–16:30 **Session IP5B: Tools and Systems**

Outilex, a Linguistic Platform for Text Processing
Olivier Blanc and Matthieu Constant

The Second Release of the RASP System
Ted Briscoe, John Carroll and Rebecca Watson

FAST – An Automatic Generation System for Grammar Tests

Chia-Yin Chen
Inst. of Info. Systems & Applications
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
G936727@oz.nthu.edu.tw

Hsien-Chin Liou
Dep. of Foreign Lang. & Lit.
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
hcliu@mx.nthu.edu.tw

Jason S. Chang
Dep. of Computer Science
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
jschang@cs.nthu.edu.tw

Abstract

This paper introduces a method for the semi-automatic generation of grammar test items by applying Natural Language Processing (NLP) techniques. Based on manually-designed patterns, sentences gathered from the Web are transformed into tests on grammaticality. The method involves representing test writing knowledge as test patterns, acquiring authentic sentences on the Web, and applying generation strategies to transform sentences into items. At runtime, sentences are converted into two types of TOEFL-style question: multiple-choice and error detection. We also describe a prototype system FAST (Free Assessment of Structural Tests). Evaluation on a set of generated questions indicates that the proposed method performs satisfactory quality. Our methodology provides a promising approach and offers significant potential for computer assisted language learning and assessment.

1 Introduction

Language testing, aimed to assess learners' language ability, is an essential part of language teaching and learning. Among all kinds of tests, grammar test is commonly used in every educational assessment and is included in well-established standardized tests like TOEFL (Test of English as Foreign Language).

Larsen-Freeman (1997) defines grammar is made of three dimensions: form, meaning, and use (See Figure 1). Hence, the goal of a grammar test is to test learners to use grammar accurately, meaningfully, and appropriately. Consider the possessive case of the personal noun in English. The possessive form comprises an apostrophe and the letter "*s*". For example, the possessive form of the personal noun "*Mary*" is "*Mary's*". The grammatical meaning of the possessive case can be (1) showing the ownership: "***Mary's book*** *is on the table.*" (= a book that belongs to Mary); (2) indicating the relationship: "***Mary's sister*** *is a student.*" (=the sister that Mary has).Therefore, a comprehensive grammar question needs to examine learners' grammatical knowledge from all three aspects (morphosyntax, semantics and pragmatics).

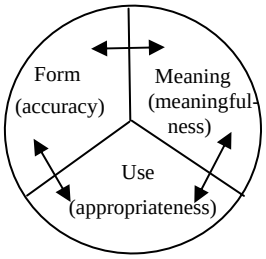


Figure 1: Three Dimensions of Grammar (Larsen-Freeman, 1997)

The most common way of testing grammar is the multiple-choice test (Kathleen and Kenji, 1996). Multiple-choice test format on grammaticality consists of two kinds: one is the traditional multiple-choice test and the other is the error detection test. Figure 2 shows a typical example of traditional multiple-choice item. As for Figure 3, it shows a sample of error detection question.

Traditional multiple-choice is composed of three components, where we define the sentence with a gap as the stem, the correct choice to the gap as the key and the other incorrect choices as the distractors. For instance, in Figure 2, the

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 1–4,
Sydney, July 2006.

partially blanked sentence acts as the stem and the key "*more than*" is accompanied by three distractors of "*as much as*", "*up as*", and "*as many to*". On the other hand, error detection item consists of a partially underlined sentence (stem) where one choice of the underlined part represents the error (key) and the other underlined parts act as distractors to distract test takers. In Figure 3, the stem is "*Although maple trees are among the most colorful varieties in the fall, they lose its leaves sooner than oak trees."* and "*its*" is the key with distractors "*among*", "*in the fall*", and "*sooner than."*

In the Great Smoky Mountains, one can see _____ 150 different kinds of tress.
(A) more than
(B) as much as
(C) up as
(D) as many to

Figure 2: An example of multiple-choice.

Although maple trees are among the most colorful
(A)
varieties in the fall, they lose its leaves
(B) (C)
sooner than oak trees.
(D)

Figure 3: An example of error detection.

Grammar tests are widely used to assess learners' grammatical competence, however, it is costly to manually design these questions. In recent years, some attempts (Coniam, 1997; Mitkov and Ha, 2003; Liu et al., 2005) have been made on the automatic generation of language testing. Nevertheless, no attempt has been made to generate English grammar tests. Additionally, previous research merely focuses on generating questions of traditional multiple-choice task, no attempt has been made for the generation of error detection test types.

In this paper, we present a novel approach to generate grammar tests of traditional multiple-choice and error detection types. First, by analyzing syntactic structure of English sentences, we constitute a number of patterns for the development of structural tests. For example, a verb-related pattern requiring an infinitive as the complement (e.g., the verb "*tend*") can be formed from the sentence "*The weather tends to improve in May.*" For each pattern, distractors are created for the completion of each grammar question. As in the case of foregoing sentence, wrong alternatives are constructed by changing the verb "*improve*" into different forms: "*to improving*", "*improve*", and "*improving.*" Then, we collect authentic sentences from the Web as the source of the tests. Finally, by applying different generation strategies, grammar tests in two test formats are produced. A complete grammar question is generated as shown in Figure 4. Intuitively, based on certain surface pattern (See Figure 5), computer is able to compose a grammar question presented in Figure 4. We have implemented a prototype system *FAST* and the experiment results have shown that about 70 test patterns can be successfully written to convert authentic Web-based texts into grammar tests.

I intend _____ you that we cannot approve your application.
(A) to inform
(B) to informing
(C) informing
(D) inform

Figure 4: An example of generated question.

* X/INFINITIVE * CLAUSE.
→
* _____* CLAUSE.
(A) X/INFINITIVE
(B) X/to VBG
(C) X/VBG
(D) X/VB

Figure 5: An example of surface pattern.

2 Related Work

Since the mid 1980s, item generation for test development has been an area of active research. In our work, we address an aspect of CAIG (computer-assisted item generation) centering on the semi-automatic construction of grammar tests.

Recently, NLP (Natural Language Processing) has been applied in CAIG to generate tests in multiple-choice format. Mitkov and Ha (2003) established a system which generates reading comprehension tests in a semi-automatic way by using an NLP-based approach to extract key concepts of sentences and obtain semantically alternative terms from WordNet.

Coniam (1997) described a process to compose vocabulary test items relying on corpus word frequency data. Recently, Gao (2000) presented a system named AWETS that semi-automatically constructs vocabulary tests based on word frequency and part-of-speech information. Most recently, Hoshino and Nakagawa (2005) established a real-time system which automatically generates vocabulary questions by utilizing machine learning techniques. Brown, Frishkoff, and Eskenazi (2005) also introduced a method on the automatic generation of 6 types of vocabulary questions by employing data from WordNet.

Liu, Wang, Gao, and Huang (2005) proposed ways of the automatic composing of English cloze items by applying word sense disambiguation method to choose target words of certain sense and collocation-based approach to select distractors.

Previous work emphasizes the automatic generation of reading comprehension, vocabulary, and cloze questions. In contrast, we present a system that allows grammar test writers to represent common patterns of test items and distractors. With these patterns, the system automatically gathers authentic sentences and generates grammar test items.

3 The *FAST* System

The question generation process of the *FAST* system includes manual design of test patterns (including construct pattern and distractor generation pattern), extracting sentences from the Web, and semi-automatic generation of test items by matching sentences against patterns. In the rest of this section, we will thoroughly describe the generation procedure.

3.1 Question Generation Algorithm

Input: *P* = common patterns for grammar test items, URL = a Web site for gathering sentences
Output: *T*, a set of grammar test items *g*

1. Crawl the site URL for webpages
2. Clean up HTML tags. Get sentences S therein that are self-contained.
3. Tag each word in *S* with part of speech (POS) and base phrase (or chunk). (See Figure 6 for the example of the tagging sentence "*A nuclear weapon is a weapon that derives its and or fusion.*")

Lemmatization: a nuclear weapon be a weapon that derive its energy from the nuclear reaction of fission and or fusion.
POS: a/at nuclear/jj weapon/nn be/bez a/at weapon/nn that/wps derive/vbz its/pp$ energy/nn from/in the/at nuclear/jj reaction/nns of/in fission/nn and/cc or/cc fusion/nn ./.
Chunk: a/B-NP nuclear/I-NP weapon/I-NP be/B-VP a/B-NP weapon/I-NP that/B-NP derive/B-VP its/B-NP energy/I-NP from/B-PP the/B-NP nuclear/I-NP reaction/I-NP of/B-PP fission/B-NP and/O or/B-UCP fusion/B-NP ./O

Figure 6: Lemmatization, POS tagging and chunking of a sentence.

4. Match *P* against *S* to get a set of candidate sentences *D*.
5. Convert each sentence *d* in *D* into a grammar test item *g*.

3.2 Writing Test Patterns

Grammar tests usually include a set of patterns covering different grammatical categories. These patterns are easily to conceptualize and to write down. In the first step of the creation process, we design test patterns.

A construct pattern can be observed through analyzing sentences of similar structural features. Sentences "*My friends enjoy traveling by plane*." and "*I enjoy surfing on the Internet*." are analyzed as an illustration. Two sentences share identical syntactic structure {* enjoy X/Gerund *}, indicating the grammatical rule for the verb "*enjoy*" needing a gerund as the complement. Similar surface patterns can be found when replacing "*enjoy*" by verbs such as "*admit*" and "*finish*" (e.g., {* admit X/Gerund *} and {* finish X/Gerund *}). These two generalize these surface patterns, we write a construct pattern {* VB VBG *} in terms of POS tags produced by a POS tagger. Thus, a construct pattern characterizing that some verbs require a gerund in the complement is contrived.

Distractor generation pattern is dependent on each designed construct pattern and therefore needs to design separately. Distractors are usually composed of words in the construct pattern with some modifications: changing part of speech, adding, deleting, replacing, or reordering of words. By way of example, in the sentence "*Strauss finished writing two of his published compositions before his tenth birthday.*", "*writing*" is the pivot word according to the construct pattern {* VBD VBG *}. Distractors for this question are: "*write*", "*written*", and "*wrote*". Similar to the way for the construct pattern devise, we use POS tags to represent distractor generation pattern: {VB}, {VBN}, and {VBD}. We define a notation scheme for the distractor designing. The symbol $0 designates the changing of the pivot word in the construct pattern while $9 and $1 are the words proceeding and following the pivot word, respectively. Hence, distractors for the abovementioned question are {$0 VB}, {$0 VBN}, and {$0 VBD}

3.3 Web Crawl for Candidate Sentences

As the second step, we extract authentic materials from the Web for the use of question stems. We collect a large number of sentences from websites containing texts of learned genres (e.g., textbook, encyclopedia).

3.4 Test Strategy

The generation strategies of multiple-choice and error detection questions are different. The generation strategy of traditional multiple-choice questions involves three steps. The first step is to empty words involved in the construct pattern. Then, according to the distractor generation pattern, three erroneous statements are produced. Finally, option identifiers (e.g., A, B, C, D) are randomly assigned to each alternative.

The test strategy for error detection questions is involved with: (1) locating the target point, (2) replacing the construct by selecting wrong statements produced based on distractor generation pattern, (3) grouping words of same chunk type to phrase chunk (e.g., "the/B-NP nickname/I-NP" becomes "the nickname/NP") and randomly choosing three phrase chunks to act as distractors, and (4) assigning options based on position order information.

4 Experiment and Evaluation Results

In the experiment, we first constructed test patterns by adapting a number of grammatical rules organized and classified in "How to Prepare for the TOEFL", a book written by Sharpe (2004). We designed 69 test patterns covering nine grammatical categories. Then, the system extracted articles from two websites, Wikipedia (an online encyclopedia) and VOA (Voice of American). Concerning about the readability issue (Dale-Chall, 1995) and the self-contained characteristic of grammar question stems, we extracted the first sentence of each article and selected sentences based on the readability distribution of simulated TOEFL tests. Finally, the system matched the tagged sentences against the test patterns. With the assistance of the computer, 3,872 sentences are transformed into 25,906 traditional multiple-choice questions while 2,780 sentences are converted into 24,221 error detection questions.

A large amount of verb-related grammar questions were blindly evaluated by seven professor/students from the TESOL program. From a total of 1,359 multiple-choice questions, 77% were regarded as 'worthy' (i.e., can be direct use or only needed minor revision) while 80% among 1,908 error detection tasks were deemed to be 'worthy'. The evaluation results indicate a satisfactory performance of the proposed method.

5 Conclusion

We present a method for the semi-automatic generation of grammar tests in two test formats by using authentic materials from the Web. At runtime, a given sentence sharing classified construct patterns is generated into tests on grammaticality. Experimental results assess the facility and appropriateness of the introduced method and indicate that this novel approach does pave a new way of CAIG.

References

Coniam, D. (1997). A Preliminary Inquiry into Using Corpus Word Frequency Data in the Automatic Generation of English Cloze Tests. CALICO Journal, No 2-4, pp. 15- 33.

Gao, Z.M. (2000). AWETS: An Automatic Web-Based English Testing System. In Proceedings of the 8th Conference on Computers in Education/International Conference on Computer-Assisted Instruction ICCE/ICCAI, 2000, Vol. 1, pp. 28-634.

Hoshino, A. & Nakagawa, H. (2005). A Real-Time Multiple-Choice Question Generation for Language Testing-A Preliminary Study-. In Proceedings of the Second Workshop on Building Educational Applications Using NLP, pp. 1-8, Ann Arbor, Michigan, 2005.

Larsen-Freeman, D. (1997). Grammar and its teaching: Challenging the myths (ERIC Digest). Washington, DC: ERIC Clearinghouse on languages and Linguistics, Center for Applied Linguistics. Retrieved July 13, 2005, from http://www.vtaide.com/png/ERIC/Grammar.htm

Liu, C.L., Wang, C.H., Gao, Z.M., & Huang, S.M. (2005). Applications of Lexical Information for Algorithmically Composing Multiple-Choice Cloze Items, In Proceedings of the Second Workshop on Building Educational Applications Using NLP, pp. 1-8, Ann Arbor, Michigan, 2005.

Mitkov, R. & Ha, L.A. (2003). Computer-Aided Generation of Multiple-Choice Tests. In Proceedings of the HLT-NAACL 2003 Workshop on Building Educational Applications Using Natural Language Processing, Edmonton, Canada, May, pp. 17 – 22.

Sharpe, P.J. (2004). How to Prepare for the TOEFL. Barrons' Educational Series, Inc.

Chall, J.S. & Dale, E. (1995). Readability Revisited: The New Dale-Chall Readability Formula. Cambridge, MA:Brookline Books.

Is It Correct? - Towards Web-Based Evaluation of Automatic Natural Language Phrase Generation

Calkin S. Montero and **Kenji Araki**
Graduate School of Information Science and Technology, Hokkaido University,
Kita 14-jo Nishi 9-chome, Kita-ku, Sapporo, 060-0814 Japan
{calkin,araki}@media.eng.hokudai.ac.jp

Abstract

This paper describes a novel approach for the automatic generation and evaluation of a *trivial dialogue phrases* database. A trivial dialogue phrase is defined as an expression used by a chatbot program as the answer of a user input. A transfer-like genetic algorithm (GA) method is used to generating the trivial dialogue phrases for the creation of a natural language generation (NLG) knowledge base. The automatic evaluation of a generated phrase is performed by producing n-grams and retrieving their frequencies from the World Wide Web (WWW). Preliminary experiments show very positive results.

1 Introduction

Natural language generation has devoted itself to studying and simulating the production of written or spoken discourse. From the *canned text* approach, in which the computer prints out a text given by a programmer, to the *template filling* approach, in which predetermined templates are filled up to produce a desired output, the applications and limitations of language generation have been widely studied. Well known applications of natural language generation can be found in human-computer conversation (HCC) systems. One of the most famous HCC systems, ELIZA (Weizenbaum, 1966), uses the template filling approach to generate the system's response to a user input. For a dialogue system, the template filling approach works well in certain situations, however due to the templates limitations, nonsense is produced easily.

In recent research Inui et al. (2003) have used a corpus-based approach to language generation. Due to its flexibility and applicability to open domain, such an approach might be considered as more robust than the template filling approach when applied to dialogue systems. In their approach, Inui et al. (2003), applied keyword matching in order to extract sample dialogues from a *dialogue corpus*, i.e., utterance-response pairs. After applying certain *transfer or exchange rules*, the sentence with maximum occurrence probability is given to the user as the system's response. Other HCC systems, e.g. Wallace (2005), have applied the corpus based approach to natural language generation in order to retrieve system's trivial dialogue responses. However, the creation of the hand crafted knowledge base, that is to say, a dialogue corpus, is a highly time consuming and hard to accomplish task[1]. Therefore we aim to automatically generate and evaluate a database of trivial dialogue phrases that could be implemented as knowledge base language generator for open domain dialogue systems, or chatbots.

In this paper, we propose the automatic generation of trivial dialogue phrases through the application of a transfer-like genetic algorithm (GA) approach. We propose as well, the automatic evaluation of the *correctness*[2] of the generated phrase using the WWW as a knowledge database. The generated database could serve as knowledge base to automatically improve publicly available chatbot[3] databases, e.g. Wallace (2005).

[1] The creation of the ALICE chatbot database (ALICE brain) has cost more that 30 researchers, over 10 years work to accomplish. http://www.alicebot.org/superbot.html http://alicebot.org/articles/wallace/dont.html

[2] Correctness implies here whether the expression is grammatically correct, and whether the expression *exists* in the Web.

[3] Computer program that simulates human conversation.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 5–8,
Sydney, July 2006.

2 Overview and Related Work

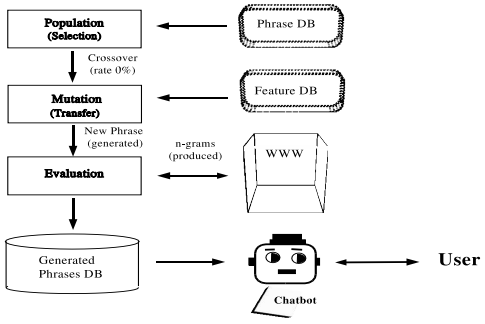


Figure 1: System Overview

We apply a GA-like transfer approach to automatically generate new trivial dialogue phrases, where each phrase is considered as a gene, and the words of the phrase represent the DNA. The transfer approach to language generation has been used by Arendse (1998), where a sentence is being *re-generated* through word substitution. Problems of erroneous grammar or ambiguity are solved by referring to a lexicon and a grammar, re-generating substitutes expressions of the original sentence, and the user deciding which one of the generated expressions is correct. Our method differs in the application of a GA-like transfer process in order to automatically insert new features on the selected original phrase and the automatic evaluation of the newly generated phrase using the WWW. We assume the automatically generated trivial phrases database is desirable as a knowledge base for open domain dialogue systems. Our system general overview is shown in Figure 1. A description of each step is given hereunder.

3 Trivial Dialogue Phrases Generation: Transfer-like GA Approach

3.1 Initial Population Selection

In the population selection process a small population of phrases are selected randomly from the Phrase DB[4]. This is a small database created beforehand. The Phrase DB was used for setting the thresholds for the evaluation of the generated phrases. It contains phrases extracted from real human-human trivial dialogues (obtained from the corpus of the University of South California (2005)) and from the hand crafted ALICE database. For the experiments this DB contained 15 trivial dialogue phrases. Some of those trivial dialogue phrases are: *do you like airplanes ?, have you have your lunch ?, I am glad you are impressed, what are your plans for the weekend ?*, and so forth. The initial population is formed by a number of phrases randomly selected between one and the total number of expressions in the database. No evaluation is performed to this initial population.

3.2 Crossover

Since the length, i.e., number of words, among the analyzed phrases differs and our algorithm does not use semantical information, in order to avoid the distortion of the original phrase, in our system the crossover rate was selected to be 0%. This is in order to ensure a language independent method. The generation of the new phrase is given solely by the mutation process explained below.

3.3 Mutation

During the mutation process, each one of the phrases of the selected initial population is mutated at a rate of $1/N$, where N is the total number of words in the phrase. The mutation is performed through a transfer process, using the Features DB. This DB contains descriptive features of different topics of human-human dialogues. The word "features" refers here to the specific part of speech used, that is, nouns, adjectives and adverbs[5]. In order to extract the descriptive features that the Feature DB contains, different human-human dialogues, (USC, 2005), were clustered by topic[6] and the most descriptive nouns, adjectives and adverbs of each topic were extracted. The word to be replaced within the original phrase is randomly selected as well as it is randomly selected the substitution feature to be used as a replacement from the Feature DB. In order to obtain a language independent system, at this stage part of speech tagging was not performed[7]. For this mutation process, the total number of possible different expressions that could be generated from a given phrase is $N^{F_{DB}}$, where the exponent F_{DB} is the total number of features in the Feature DB.

[4] In this paper DB stands for database.

[5] For the preliminary experiment this database contained 30 different features

[6] Using agglomerative clustering with the publicly available Cluto toolkit

[7] POS tagging was used when creating the Features DB. Alternatively, instead of using POS, the features might be given by hand

Total no Phrases Gen		Unnatural		Usable		Completely Natural		Precision	Recall
Accepted	Rejected	Accepted	Rejected	Accepted	Rejected	Accepted	Rejected		
80	511	36	501	18	8	26	2	0.550	0.815
Total 591		Total 537		Total 26		Total 28			

Table 3. Human Evaluation - Naturalness of the Phrases

3.4 Evaluation

In order to evaluate the correctness of the newly generated expression, we used as database the WWW. Due to its significant growth[8], the WWW has become an attractive database for different systems applications as, machine translation (Resnik and Smith, 2003), question answering (Kwok et al., 2001), commonsense retrieval (Matuszek et al., 2005), and so forth. In our approach we attempt to evaluate whether a generated phrase is correct through its frequency of appearance in the Web, i.e., the *fitness* as a function of the frequency of appearance. Since matching an entire phrase on the Web might result in very low retrieval, in some cases even non retrieval at all, we applied the sectioning of the given phrase into its respective n-grams.

3.4.1 N-Grams Production

For each one of the generated phrases to evaluate, n-grams are produced. The n-grams used are bigram, trigram, and quadrigram. Their frequency of appearance on the Web (using Google search engine) is searched and ranked. For each n-gram, thresholds have been established[9]. A phrase is evaluated according to the following algorithm[10]:

`if` $\alpha < NgramFreq < \theta$, `then` $Ngram$ "weakly accepted"
`elsif` $NgramFreq > \theta$, `then` $Ngram$ "accepted"
`else` $Ngram$ "rejected"

where, α and θ are thresholds that vary according to the n-gram type, and $NgramFreq$ is the frequency, or number of hits, returned by the search engine for a given n-gram. Table 1 shows some of the n-grams produced for the generated phrase "what are your plans for the game?" The frequency of each n-gram is also shown along with the system evaluation. The phrase was evaluated as *accepted* since none of the n-grams produced was *rejected*.

	N-Gram	Frequency (hits)	System Eval.
Bigram	what:are	213000000	accepted
Trigram	your:plans:for	116000	accepted
Quadrigram	plans:for:the:game	958	accepted

Table 1. N-Grams Produced for:
"what are your plans for the game?"

4 Preliminary Experiments and Results

The system was setup to perform 150 generations[11]. Table 2 contains the results. There were 591 different phrases generated, from which 80 were evaluated as "accepted", and the rest 511 were rejected by the system.

Total Generations	150
Total Generated Phrases	591
Accepted	80
Rejected	511

Table 2. Results for 150 Generations

As part of the preliminary experiment, the generated phrases were evaluated by a native English speaker in order to determine their "naturalness". The human evaluation of the generated phrases was performed under the criterion of the following categories:

a) Unnatural: a phrase that would not be used during a conversation.

b) Usable: a phrase that could be used during a conversation,even though it is not a common phrase.

c) Completely Natural: a phrase that might be commonly used during a conversation.

The results of the human evaluation are shown in Table 3. In this evaluation, 26 out of the 80 phrases "accepted" by the system were considered "completely natural", and 18 out of the 80 "accepted" were considered "usable", for a total of 44 *well-generated* phrases[12]. On the other hand, the system mis-evaluation is observed mostly within the "accepted" phrases, i.e., 36 out of 80 "accepted" were "unnatural", whereas within the "rejected" phrases only 8 out of 511 were considered "usable" and 2 out of 511 were considered "completely natural", which affected negatively the pre-

[8] As for 1998, according to Lawrence and Giles (1999) the "surface Web" consisted of approximately 2.5 billion documents. As for January 2005, according to Gulli and Signorini (2005),the size of indexable Web had become approximately 11.5 billion pages

[9] The tuning of the thresholds of each n-gram type was preformed using the phrases of the Phrase DB

[10] The evaluation "weakly accepted" has been designed to reflect n-grams whose appearance on the Web is significant even though they are rarely used. In the experiment they were treated as *accepted*.

[11] Processing time: 20 hours 13 minutes. The Web search results are as for March 2006

[12] Phrases that could be used during a conversation

Original Phrase	Generated Phrase
what are your plans for the weekend ?	**Completely Natural**
	what are your plans for the game ?
	Usable
	what are your friends for the weekend ?
	Unnatural
	what are your plans for the visitation ?

Table 4. Examples of Generated Phrases

cision of the system.

In order to obtain a statistical view of the system's performance, the metrics of recall, (*R*), and precision, (*P*), were calculated according to (*A* stands for "Accepted", from Table 3):

$$R = \frac{Usable(A)+CompletelyNatural(A)}{UsableTotal+CompletelyNaturalTotal}$$

$$P = \frac{Usable(A)+CompletelyNatural(A)}{Unnatural(A)+Usable(A)+CompletelyNatural(A)}$$

Table 4 shows the system output, i.e., phrases generated and evaluated as "accepted" by the system, for the original phrase "what are your plans for the weekend ?" According with the criterion shown above, the generated phrases were evaluated by a user to determine their naturalness - applicability to dialogue.

4.1 Discussion

Recall is the rate of the *well-generated* phrases given as "accepted" by the system divided by the *total number* of well-generated phrases. This is a measure of the coverage of the system in terms of the well-generated phrases. On the other hand, the precision rates the well-generated phrases divided by the total number of "accepted" phrases. The precision is a measure of the correctness of the system in terms of the evaluation of the phrases. For this experiment the recall of the system was 0.815, i.e., 81.5% of the total number of well-generated phrases where correctly selected, however this implied a trade-off with the precision, which was compromised by the system's wide coverage.

An influential factor in the system precision and recall is the selection of new features to be used during the mutation process. This is because the insertion of a new feature gives rise to a totally new phrase that might not be related to the original one. In the same tradition, a decisive factor in the evaluation of a well-generated phrase is the constantly changing information available on the Web. This fact rises thoughts of the application of *variable* threshold for evaluation. Even though the system leaves room for improvement, its successful implementation has been confirmed.

5 Conclusions and Future Directions

We presented an automatic trivial dialogue phrases generator system. The generated phrases are automatically evaluated using the frequency hits of the n-grams correspondent to the analyzed phrase. However improvements could be made in the evaluation process, preliminary experiments showed a promising successful implementation. We plan to work toward the application of the obtained database of trivial phrases to open domain dialogue systems.

References

Bernth Arendse. 1998. Easyenglish: Preprocessing for MT. In *Proceedings of the Second International Workshop on Controlled Language Applications (CLAW98)*, pages 30–41.

Antonio Gulli and Alessio Signorini. 2005. The indexable web is more than 11.5 billion pages. In *In Proceedings of 14th International World Wide Web Conference*, pages 902–903.

Nobuo Inui, Takuya Koiso, Junpei Nakamura, and Yoshiyuki Kotani. 2003. Fully corpus-based natural language dialogue system. In *Natural Language Generation in Spoken and Written Dialogue, AAAI Spring Symposium*.

Cody Kwok, Oren Etzioni, and Daniel S. Weld. 2001. Scaling question answering to the web. *ACM Trans. Inf. Syst.*, 19(3):242–262.

Steve Lawrence and Lee Giles. 1999. Accessibility of information on the web. *Nature*, 400(107-109).

Cynthia Matuszek, Michael Witbrock, Robert C. Kahlert, John Cabral, Dave Schneider, Purvesh Shah, and Doug Lenat. 2005. Searching for common sense: Populating cyc(tm) from the web. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Comput. Linguist.*, 29(3):349–380.

University of South California USC. 2005. Dialogue diversity corpus. http://www-rcf.usc.edu/˜billmann/diversity/DDivers-site.htm.

Richard Wallace. 2005. A.l.i.c.e. artificial intelligence foundation. http://www.alicebot.org.

Joseph Weizenbaum. 1966. Elizaa computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45.

LeXFlow: a System for Cross-fertilization of Computational Lexicons

Maurizio Tesconi and **Andrea Marchetti**
CNR-IIT
Via Moruzzi 1, 56024 Pisa, Italy
{maurizio.tesconi,andrea.marchetti}@iit.cnr.it
Francesca Bertagna and **Monica Monachini** and **Claudia Soria** and **Nicoletta Calzolari**
CNR-ILC
Via Moruzzi 1, 56024 Pisa, Italy
{francesca.bertagna,monica.monachini,
claudia.soria,nicoletta.calzolari}@ilc.cnr.it

Abstract

This demo presents LeXFlow, a workflow management system for cross-fertilization of computational lexicons. Borrowing from techniques used in the domain of document workflows, we model the activity of lexicon management as a set of workflow types, where lexical entries move across agents in the process of being dynamically updated. A prototype of LeXFlow has been implemented with extensive use of XML technologies (XSLT, XPath, XForms, SVG) and open-source tools (Cocoon, Tomcat, MySQL). LeXFlow is a web-based application that enables the cooperative and distributed management of computational lexicons.

1 Introduction

LeXFlow is a workflow management system aimed at enabling the semi-automatic management of computational lexicons. By management we mean not only creation, population and validation of lexical entries but also integration and *enrichment* of different lexicons.

A lexicon can be enriched by resorting to automatically acquired information, for instance by means of an application extracting information from corpora. But a lexicon can be enriched also by resorting to the information available in another lexicon, which can happen to encode different types of information, or at different levels of granularity. LeXFlow intends to address the request by the computational lexicon community for a change in perspective on computational lexicons: from static resources towards *dynamically configurable multi-source entities*, where the content of lexical entries is dynamically modified and updated on the basis of the integration of knowledge coming from different sources (indifferently represented by human actors, other lexical resources, or applications for the automatic extraction of lexical information from texts).

This scenario has at least two strictly related prerequisites: i) existing lexicons have to be available in or be mappable to a standard form enabling the overcoming of their respective differences and idiosyncrasies, thus making their mutual comprehensibility a reality; ii) an architectural framework should be used for the effective and practical management of lexicons, by providing the communicative channel through which lexicons can really communicate and share the information encoded therein.

For the first point, standardization issues obviously play the central role. Important and extensive efforts have been and are being made towards the extension and integration of existing and emerging open lexical and terminological standards and best practices, such as EAGLES, ISLE, TEI, OLIF, Martif (ISO 12200), Data Categories (ISO 12620), ISO/TC37/SC4, and LIRICS. An important achievement in this respect is the MILE, a meta-entry for the encoding of multilingual lexical information (Calzolari et al., 2003); in our approach we have embraced the MILE model.

As far as the second point is concerned, some initial steps have been made to realize frameworks enabling inter-lexica access, search, integration and operability. Nevertheless, the general impression is that little has been made towards the development of new methods and techniques

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 9–12,
Sydney, July 2006.

for the concrete interoperability among lexical and textual resources. The intent of LeXFlow is to fill in this gap.

2 LeXFlow Design and Application

LeXFlow is conceived as a metaphoric extension and adaptation to computational lexicons of XFlow, a framework for the management of document workflows (DW, Marchetti et al., 2005).

A DW can be seen as a process of *cooperative authoring* where the document can be the goal of the process or just a side effect of the cooperation. Through a DW, a document life-cycle is tracked and supervised, continually providing control over the actions leading to document compilation In this environment a document travels among *agents* who essentially carry out the pipeline receive-process-send activity.

Each lexical entry can be modelled as a document instance (formally represented as an XML representation of the MILE lexical entry), whose behaviour can be formally specified by means of a document workflow type (DWT) where different agents, with clear-cut roles and responsibilities, act over different portions of the same entry by performing different tasks.

Two types of agents are envisaged: *external agents* are human or software actors which perform activities dependent from the particular DWT, and *internal agents* are software actors providing general-purpose activities useful for any DWT and, for this reason, implemented directly into the system. Internal agents perform general functionalities such as creating/converting a document belonging to a particular DWT, populating it with some initial data, duplicating a document to be sent to multiple agents, splitting a document and sending portions of information to different agents, merging duplicated documents coming from multiple agents, aggregating fragments, and finally terminating operations over the document. An external agent *executes* some processing using the document content and possibly other data, e.g. *updates* the document inserting the results of the preceding processing, *signs* the updating and finally *sends* the document to the next agent(s).

The state diagram in Figure 1 describes the different states of the document instances. At the starting point of the document life cycle there is a creation phase, in which the system raises a new instance of a document with information attached.

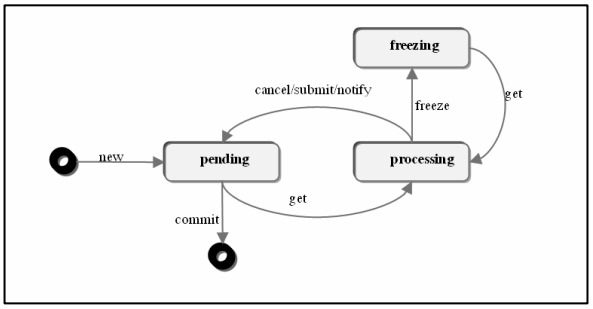


Figure 1. Document State Diagram.

The document instance goes into *pending* state. When an agent gets the document, it goes into *processing* state in which the agent compiles the parts under his/her responsibility. If the agent, for some reason, doesn't complete the instance elaboration, he can save the work performed until that moment and the document instance goes into *freezing* state. If the elaboration is completed (submitted), or cancelled, the instance goes back into *pending* state, waiting for a new elaboration.

Borrowing from techniques used in DWs, we have modelled the activity of lexicon management as a set of DWT, where lexical entries move across agents and become dynamically updated.

3 Lexical Workflow General Architecture

As already written, LeXFlow is based on XFlow which is composed of three parts: i) the Agent Environment, i.e. the agents participating to all DWs; ii) the Data, i.e. the DW descriptions plus the documents created by the DW and iii) the Engine. Figure 2 illustrates the architecture of the framework.

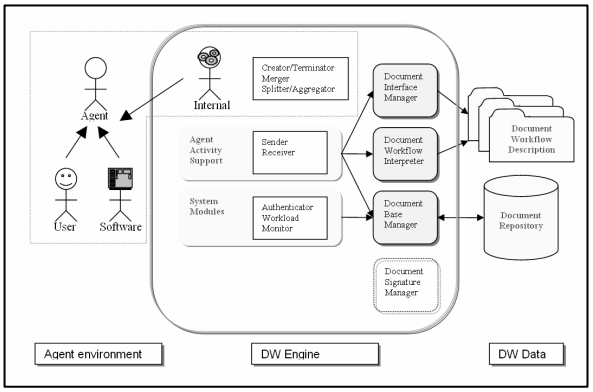


Figure 2. General Architecture.

The DW environment is the set of human and software agents participating to at least one DW.

The description of a DW can be seen as an extension of the XML document class. A class of documents, created in a DW, shares the schema of their structure, as well as the definition of the procedural rules driving the DWT and the list of the agents attending to it. Therefore, in order to describe a DWT, we need four components:

- a schema of the documents involved in the DWT;
- the agent roles chart, i.e. the set of the external and internal agents, operating on the document flow. Inside the role chart these agents are organized in roles and groups in order to define who has access to the document. This component constitutes the DW environment;
- a document interface description used by external agents to access the documents. This component also allows checking access permissions to the document;
- a document workflow description defining all the paths that a document can follow in its life-cycle, the activities and policies for each role.

The document workflow engine constitutes the run-time support for the DW, it implements the *internal agents,* the support for *agents' activities*, and some *system modules* that the external agents have to use to interact with the DW system. Also, the engine is responsible for two kinds of documents useful for each document flow: the *documents system logs* and the *documents system metadata.*

4 The lexicon Augmentation Workflow Type

In this section we present a first DWT, called "lexicon augmentation", for dynamic augmentation of semantic MILE-compliant lexicons. This DWT corresponds to the scenario where an entry of a lexicon *A* becomes enriched via basically two steps. First, by virtue of being mapped onto a corresponding entry belonging to a lexicon *B*, the $entry_{(A)}$ inherits the semantic relations available in the mapped $entry_{(B)}$. Second, by resorting to an automatic application that acquires information about semantic relations from corpora, the acquired relations are integrated into the entry and proposed to the human encoder.

In order to test the system we considered the Simple/Clips (Ruimy et al., 2003) and ItalWordNet (Roventini et al., 2003) lexicons.

An overall picture of the flow is shown in Figure 3, illustrating the different agents participating to the flow. Rectangles represent human actors over the entries, while the other figures symbolize software agents: ovals are internal agents and octagons external ones. The functionality offered to human agents are: display of MILE-encoded lexical entries, selection of lexical entries, mapping between lexical entries be-

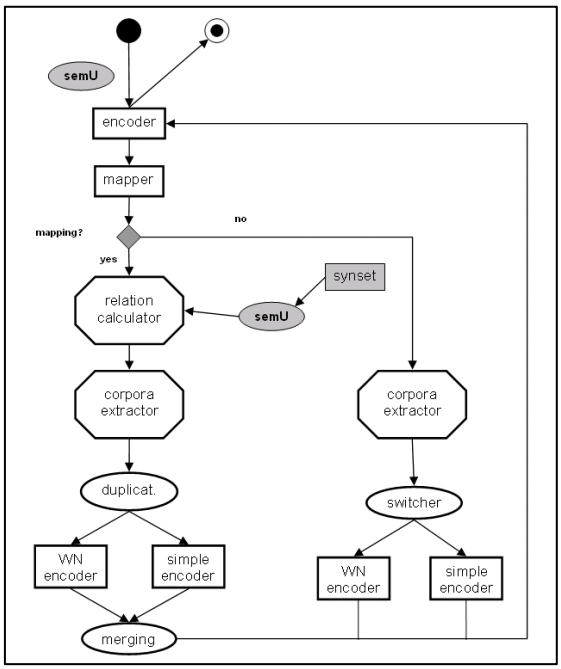


Figure 3. Lexicon Augmentation Workflow.

longing to different lexicons[1], automatic calculations of new semantic relations (either automatically derived from corpora and mutually inferred from the mapping) and manual verification of the newly proposed semantic relations.

5 Implementation Overview

Our system is currently implemented as a web-based application where the human external agents interact with system through a web browser. All the human external agents attending the different document workflows are the users of system. Once authenticated through username and password the user accesses his workload area where the system lists all his pending documents (i.e. entries) sorted by type of flow.

The system shows only the flows to which the user has access. From the workload area the user

[1] We hypothesize a human agent, but the same role could be performed by a software agent. To this end, we are investigating the possibility of automatically exploiting the procedure described in (Ruimy and Roventini, 2005).

can browse his documents and select some operations

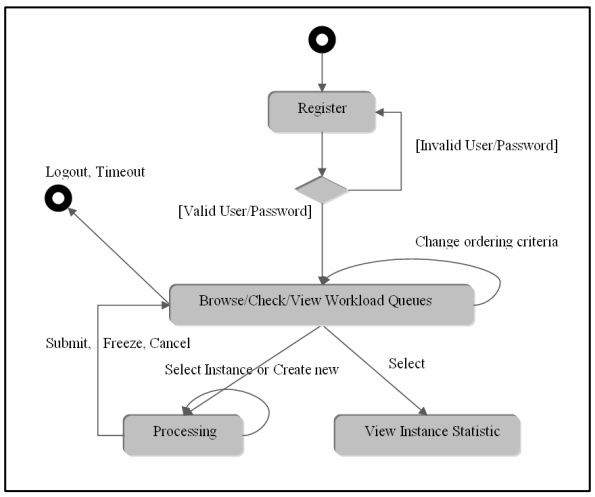


Figure 4. LeXFlow User Activity State Diagram.

such as: selecting and processing pending document; creating a new document; displaying a graph representing a DW of a previously created document; highlighting the current position of the document. This information is rendered as an SVG (Scalable Vector Graphics) image. Figure 5 illustrates the overall implementation of the system.

5.1 The Client Side: External Agent Interaction

The form used to process the documents is rendered with XForms. Using XForms, a browser can communicate with the server through XML documents and is capable of displaying the document with a user interface that can be defined for each type of document. A browser with XForms capabilities will receive an XML document that will be displayed according to the specified template, then it will let the user edit the document and finally it will send the modified document to the server.

5.2 The Server Side

The server-side is implemented with Apache Tomcat, Apache Cocoon and MySQL. Tomcat is used as the web server, authentication module (when the communication between the server and the client needs to be encrypted) and servlet container. Cocoon is a publishing framework that uses the power of XML. The entire functioning of Cocoon is based on one key concept: component pipelines. The pipeline connotes a series of events, which consists of taking a request as input, processing and transforming it, and then giving the desired response. MySQL is used for storing and retrieving the documents and the status of the documents.

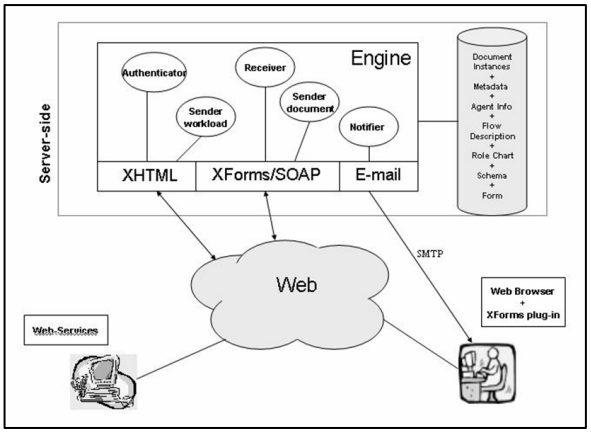


Figure 5. Overall System Implementation.

Each software agent is implemented as a web-service and the WSDL language is used to define its interface.

References

Nicoletta Calzolari, Francesca Bertagna, Alessandro Lenci and Monica Monachini, editors. 2003. *Standards and Best Practice for Multilingual Computational Lexicons. MILE (the Multilingual ISLE Lexical Entry).* ISLE Deliverable D2.2 & 3.2. Pisa.

Andrea Marchetti, Maurizio Tesconi, and Salvatore Minutoli. 2005. XFlow: An XML-Based Document-Centric Workflow. In *Proceedings of WISE'05*, pages 290- 303, New York, NY, USA.

Adriana Roventini, Antonietta Alonge, Francesca Bertagna, Nicoletta Calzolari, Christian Girardi, Bernardo Magnini, Rita Marinelli, and Antonio Zampolli. 2003. ItalWordNet: Building a Large Semantic Database for the Automatic Treatment of Italian. In Antonio Zampolli, Nicoletta Calzolari, and Laura Cignoni, editors, *Computational Linguistics in Pisa*, Istituto Editoriale e Poligrafico Internazionale, Pisa-Roma, pages 745-791.

Nilda Ruimy, Monica Monachini, Elisabetta Gola, Nicoletta Calzolari, Cristina Del Fiorentino, Marisa Ulivieri, and Sergio Rossi. 2003. A Computational Semantic Lexicon of Italian: SIMPLE. In Antonio Zampolli, Nicoletta Calzolari, and Laura Cignoni, editors, *Computational Linguistics in Pisa*, Istituto Editoriale e Poligrafico Internazionale, Pisa-Roma, pages 821-864.

Nilda Ruimy and Adriana Roventini. 2005. Towards the linking of two electronic lexical databases of Italian. In *Proceedings of L&T'05 - Language Technologies as a Challenge for Computer Science and Linguistics*, pages 230-234, Poznan, Poland.

Valido: a Visual Tool for Validating Sense Annotations

Roberto Navigli
Dipartimento di Informatica
Università di Roma "La Sapienza"
Roma, Italy
navigli@di.uniroma1.it

Abstract

In this paper we present Valido, a tool that supports the difficult task of validating sense choices produced by a set of annotators. The validator can analyse the semantic graphs resulting from each sense choice and decide which sense is more coherent with respect to the structure of the adopted lexicon. We describe the interface and report an evaluation of the tool in the validation of manual sense annotations.

1 Introduction

The task of sense annotation consists in the assignment of the appropriate senses to words in context. For each word, the senses are chosen with respect to a sense inventory encoded by a reference dictionary. The free availability and, as a result, the massive adoption of WordNet (Fellbaum, 1998) largely contributed to its status of *de facto* standard in the NLP community. Unfortunately, WordNet is a fine-grained resource, which encodes possibly subtle sense distictions.

Several studies report an inter-annotator agreement around 70% when using WordNet as a reference sense inventory. For instance, the agreement in the Open Mind Word Expert project (Chklovski and Mihalcea, 2002) was 67.3%. Such a low agreement is only in part due to the inexperience of sense annotators (e.g. volunteers on the web). Rather, to a large part it is due to the difficulty in making clear which are the real distinctions between close word senses in the WordNet inventory.

Adjudicating sense choices, i.e. the task of validating word senses, is therefore critical in building a high-quality data set. The validation task can be defined as follows: let w be a word in a sentence σ, previously annotated by a set of annotators $A = \{a_1, a_2, ..., a_n\}$ each providing a sense for w, and let $S_A = \{s_1, s_2, ..., s_m\} \subseteq Senses(w)$ be the set of senses chosen for w by the annotators in A, where $Senses(w)$ is the set of senses of w in the reference inventory (e.g. WordNet). A validator is asked to validate, that is to adjudicate a sense $s \in Senses(w)$ for a word w over the others. Notice that s is a word sense for w in the sense inventory, but is not necessarily in S_A, although it is likely to be. Also note that the annotators in A can be either human or automatic, depending upon the purpose of the exercise.

2 Semantic Interconnections

Semantic graphs are a notation developed to represent knowledge explicitly as a set of conceptual entities and their interrelationships. Fields like the analysis of the lexical text cohesion (Morris and Hirst, 1991), word sense disambiguation (Agirre and Rigau, 1996; Mihalcea and Moldovan, 2001), ontology learning (Navigli and Velardi, 2005), etc. have certainly benefited from the availability of wide-coverage computational lexicons like WordNet (Fellbaum, 1998), as well as semantically annotated corpora like SemCor (Miller et al., 1993).

Recently, a knowledge-based algorithm for Word Sense Disambiguation, called *Structural Semantic Interconnections*[1] (SSI) (Navigli and Velardi, 2004), has been shown to provide interesting insights into the choice of word senses by providing structural justifications in terms of semantic graphs.

SSI exploits an extensive lexical knowledge base, built upon the WordNet lexicon and enriched with collocation information representing seman-

[1] SSI is available online at http://lcl.di.uniroma1.it/ssi.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 13–16,
Sydney, July 2006.

tic relatedness between sense pairs. Collocations are acquired from existing resources (like the Oxford Collocations, the Longman Language Activator, collocation web sites, etc.). Each collocation is mapped to the WordNet sense inventory in a semi-automatic manner and transformed into a *relatedness* edge (Navigli and Velardi, 2005).

Given a word context $C = \{w_1, ..., w_k\}$, SSI builds a graph $G = (V, E)$ such that $V = \bigcup_{i=1}^{k} Senses_{\text{WN}}(w_i)$ and $(s, s') \in E$ if there is at least one semantic interconnection between s and s' in the lexical knowledge base. A *semantic interconnection pattern* is a relevant sequence of edges selected according to a manually-created context-free grammar, i.e. a path connecting a pair of word senses, possibly including a number of intermediate concepts. The grammar consists of a small number of rules, inspired by the notion of lexical chains (Morris and Hirst, 1991). An excerpt of the context-free grammar encoding semantic interconnection patterns for the WordNet lexicon is reported in Table 1. For the full set of interconnections the reader can refer to Navigli and Velardi (2004).

SSI performs disambiguation in an iterative fashion, by maintaining a set $\mathcal{C}$ of senses as a semantic context. Initially, $\mathcal{C} = V$ (the entire set of senses of words in C). At each step, for each sense s in C, the algorithm calculates a score of the degree of connectivity between s and the other senses in $\mathcal{C}$:

$$Score_{SSI}(s, \mathcal{C}) = \frac{\sum_{s' \in \mathcal{C} \setminus \{s\}} \sum_{i \in IC(s,s')} \frac{1}{length(i)}}{\sum_{s' \in \mathcal{C} \setminus \{s\}} |IC(s,s')|}$$

where $IC(s, s')$ is the set of interconnections between senses s and s'. The contribution of a single interconnection is given by the reciprocal of its length, calculated as the number of edges connecting its ends. The overall degree of connectivity is then normalized by the number of contributing interconnections. The highest ranking sense s of word w is chosen and the senses of w are removed from the semantic context $\mathcal{C}$. The algorithm terminates when either $\mathcal{C} = \emptyset$ or there is no sense such that its score exceeds a fixed threshold.

3 The Tool: Valido

Based on SSI, we developed a visual tool, *Valido*[2], to visually support the validator in the difficult task of assessing the quality and suitability of sense annotations. The tool takes as input a corpus of documents whose sentences were previously tagged by one or more annotators with word senses from the WordNet inventory. The corpus can be input in xml format, as specified in the initial page.

[2]Valido is available at http://lcl.di.uniroma1.it/valido.

$S \rightarrow S'S_1 \mid S'S_2 \mid S'S_3$ (start rule)
$S' \rightarrow e_{nominalization} \mid e_{pertainymy} \mid \epsilon$ (part-of-speech jump)
$S_1 \rightarrow e_{kind-of}S_1 \mid e_{part-of}S_1 \mid e_{kind-of} \mid e_{part-of}$ (hyperonymy/meronymy)
$S_2 \rightarrow e_{kind-of}S_2 \mid e_{relatedness}S_2 \mid e_{kind-of} \mid e_{relatedness}$ (hypernymy/relatedness)
$S_3 \rightarrow e_{similarity}S_3 \mid e_{antonymy}S_3 \mid e_{similarity} \mid e_{antonymy}$ (adjectives)

Table 1: An excerpt of the context-free grammar for the recognition of semantic interconnections.

The user can browse the sentences, and adjudicate a choice over the others in case of disagreement among the annotators. To the end of assisting the user in the validation task, the tool highlights each word in a sentence with different colors, namely: *green* for words having a full agreement, *red* for words where no agreement can be found, *orange* for those words on which a validation policy can be applied.

A validation policy is a strategy for suggesting a default sense choice to the validator in case of disagreement. Initially, the validator can choose one of four validation policies to be applied to those words with disagreement on which sense to assign:

(α) **majority voting**: if there exists a sense $s \in S_A$ (the set of senses chosen by the annotators in A) such that $\frac{|\{a \in A \mid a \text{ annotated } w \text{ with } s\}|}{|A|} \geq \frac{1}{2}$, s is proposed as the preferred sense for w;

(β) **majority voting + SSI**: the same as the previous policy, with the addition that if there exists no sense chosen by a majority of annotators, SSI is applied to w, and the sense chosen by the algorithm, if any, is proposed to the validator;

(γ) **SSI**: the SSI algorithm is applied to w, and the chosen sense, if any, is proposed to the validator;

(δ) **no validation**: w is left untagged.

Notice that for policies (β) and (γ) Valido applies the SSI algorithm to w in the context of its

sentence σ by taking into account for disambiguation only the senses in s (i.e. the set of senses chosen by the annotators). In general, given a set of words with disagreement $W \subseteq \sigma$, SSI is applied to W using as a fixed context the agreed senses chosen for the words in $\sigma \setminus W$.

Also note that the suggestion of a sense choice, marked in orange based on the validation policy, is just a proposal and can freely modified by the validator, as explained hereafter.

Before starting the interface, the validator can also choose whether to add a virtual annotator a_{SSI} to the set of annotators A. This virtual annotator tags each word $w \in \sigma$ with the sense chosen by the application of the SSI algorithm to σ. As a result, the selected validation policy will be applied to the new set of annotators $A' = A \cup \{a_{SSI}\}$. This is useful especially when $|A| = 1$ (e.g. in the automatic application of a single word sense disambiguation system), that is when validation policies are of no use.

Figure 1 illustrates the interface of the tool: in the top pane the sentence at hand is shown, marked with colors as explained above. The main pane shows the semantic interconnections between senses for which either there is a full agreement or the chosen validation policy can be applied. When the user clicks on a word w, the left pane reports the sense inventory for w, including information about the hypernym, definition and usage for each sense of w. The validator can then click on a sense and see how the semantic graph shown in the main pane changes after the selection, possibly resulting in a different number and strength of semantic interconnection patterns supporting that sense choice. For each sense in the left pane, the annotators in A who favoured that choice are listed (for instance, in the figure annotator #1 chose sense #1 of *street*, while annotator #2 as well as SSI chose sense #2).

If the validator decides that a certain word sense is more convincing based on its semantic graph, (s)he can select that sense as a final choice by clicking on the *validate* button on top of the left pane. In case the validator wants to validate present sense choices of all the disagreed words, (s)he can press the *validate all* button in the top pane. As a result, the present selection of senses will be chosen as the final configuration for the entire sentence at hand.

In the top pane, an icon beside each disagreed word shows the validation status of the word: a *question mark* indicates that the disagreement has not yet been solved, while a *checkmark* indicates that the validator solved the disagremeent.

	Precision	**Recall**
Nouns	75.80% (329/434)	63.75% (329/516)
Adjectives	74.19% (46/62)	22.33% (46/206)
Verbs	65.64% (107/163)	43.14% (107/248)
Total	73.14% (482/659)	49.69% (482/970)

Table 2: Results on 1,000 sentences from SemCor.

4 Evaluation

We briefly report here an experiment on the validation of manual sense annotations with the aid of Valido. For more detailed experiments the reader can refer to Navigli (2006).

1,000 sentences were uniformly selected from the set of documents in the semantically-tagged SemCor corpus (Miller et al., 1993). For each sentence $\sigma = w_1 w_2 \dots w_k$ annotated in SemCor with the senses $s_{w_1} s_{w_2} \dots s_{w_k}$ ($s_{w_i} \in Senses(w_i), i \in \{1, 2, \dots, k\}$), we randomly identified a word $w_i \in \sigma$, and chose at random a different sense $\overline{s}_{w_i}$ for that word, that is $\overline{s}_{w_i} \in Senses(w_i) \setminus \{s_{w_i}\}$. In other words, we simulated *in vitro* a situation in which an annotator provides an appropriate sense and the other selects a different sense.

We applied Valido with policy (γ) to the annotated sentences and evaluated the performance of the approach in suggesting the appropriate choice for the words with disagreement. The results are reported in Table 2 for nouns, adjectives, and verbs (we neglected adverbs as very few interconnections can be found for them).

The experiment shows that evidences of inconsistency due to inappropriate annotations are provided with good precision. The overall F1 measure is 59.18%. The chance baseline is 50%.

The low recall obtained for verbs, but especially for adjectives, is due to a lack of connectivity in the lexical knowledge base, when dealing with connections across different parts of speech.

5 Conclusions

In this paper we presented Valido, a tool for the validation of manual and automatic sense annotations. Valido allows a validator to analyse the coherency of different sense annotations provided for the same word in terms of the respective semantic interconnections with the other senses in context. We reported an experiment showing that

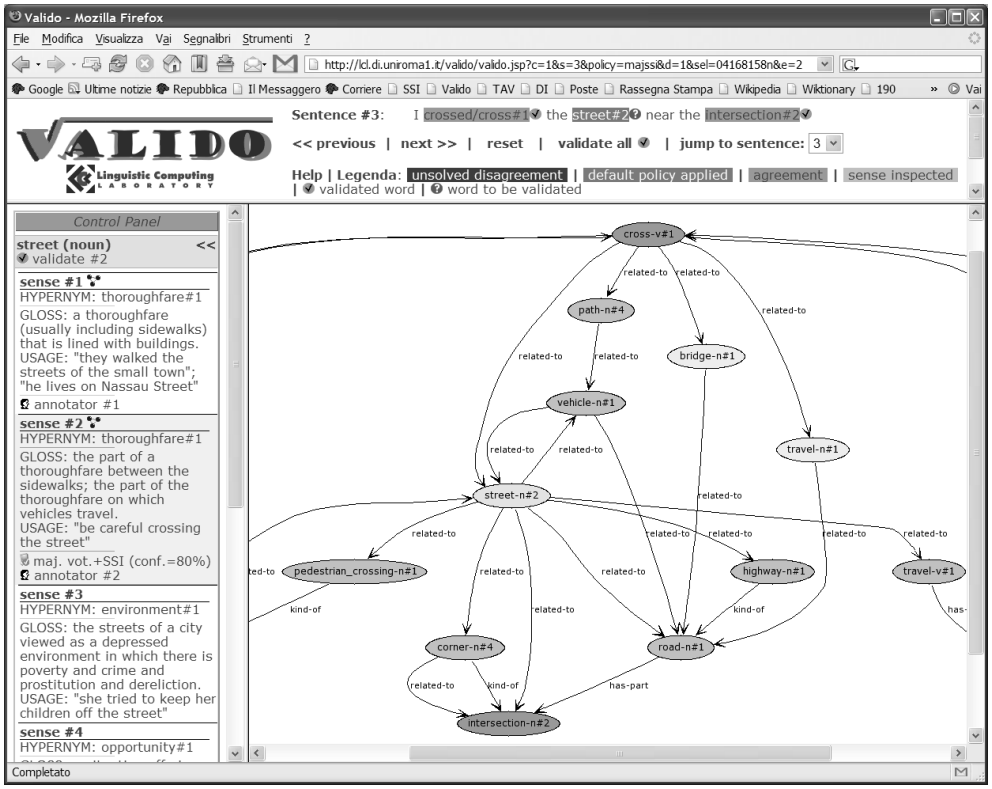


Figure 1: A screenshot of the tool.

the approach provides useful hints. Notice that this experiment concerns the quality of the suggestions, which are not necessarily taken into account by the validator (implying a higher degree of accuracy in the overall validation process).

We foresee an extension of the tool for supporting the sense annotation phase. The tool can indeed provide richer information than interfaces like the Open Mind Word Expert (Chklovski and Mihalcea, 2002), and the annotator can take advantage of the resulting graphs to improve awareness in the decisions to be taken, so as to make consistent choices with respect to the reference lexicon.

Finally, we would like to propose the use of the tool in the preparation of at least one of the test sets for the next Senseval exercise, to be held supposedly next year.

Acknowledgments

This work is partially funded by the Interop NoE (508011), 6^{th} European Union FP.

References

Eneko Agirre and German Rigau. 1996. Word sense disambiguation using conceptual density. In *Proc. of COLING 1996*. Copenhagen, Denmark.

Tim Chklovski and Rada Mihalcea. 2002. Building a sense tagged corpus with open mind word expert. In *Proc. of ACL 2002 Workshop on WSD: Recent Successes and Future Directions*. Philadelphia, PA.

Christiane Fellbaum, editor. 1998. *WordNet: an Electronic Lexical Database*. MIT Press.

Rada Mihalcea and Dan Moldovan. 2001. Automatic generation of a coarse grained wordnet. In *Proc. of NAACL Workshop on WordNet and Other Lexical Resources*. Pittsburgh, PA.

George Miller, Claudia Leacock, Tengi Randee, and Ross Bunker. 1993. A semantic concordance. In *Proc. 3^{rd} DARPA Workshop on Human Language Technology*. Plainsboro, New Jersey.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1).

Roberto Navigli and Paola Velardi. 2004. Learning domain ontologies from document warehouses and dedicated websites. *Computational Linguistics*, 30(2).

Roberto Navigli and Paola Velardi. 2005. Structural semantic interconnections: a knowledge-based approach to word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(7).

Roberto Navigli. 2006. Experiments on the validation of sense annotations assisted by lexical chains. In *Proc. of the European Chapter of the Annual Meeting of the Association for Computational Linguistics (EACL)*. Trento, Italy.

An intelligent search engine and GUI-based efficient MEDLINE search tool based on deep syntactic parsing

Tomoko Ohta **Yusuke Miyao** **Takashi Ninomiya**[¶]
Yoshimasa Tsuruoka[*†] **Akane Yakushiji**[‡] **Katsuya Masuda**
Jumpei Takeuchi **Kazuhiro Yoshida** **Tadayoshi Hara**
Jin-Dong Kim **Yuka Tateisi**[§] **Jun'ichi Tsujii**
Department of Computer Science, University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033 JAPAN
{okap, yusuke, ninomi, tsuruoka, akane, kmasuda, tj_jug,
kyoshida, harasan, jdkim, yucca, tsujii}@is.s.u-tokyo.ac.jp

Abstract

We present a practical HPSG parser for English, an intelligent search engine to retrieve MEDLINE abstracts that represent *biomedical events* and an efficient MEDLINE search tool helping users to find information about biomedical entities such as *genes*, *proteins*, and the *interactions* between them.

1 Introduction

Recently, biomedical researchers have been facing the vast repository of research papers, e.g. MEDLINE. These researchers are eager to search biomedical correlations such as protein-protein or gene-disease associations. The use of natural language processing technology is expected to reduce their burden, and various attempts of information extraction using NLP has been being made (Blaschke and Valencia, 2002; Hao et al., 2005; Chun et al., 2006). However, the framework of traditional information retrieval (IR) has difficulty with the accurate retrieval of such relational concepts. This is because relational concepts are essentially determined by semantic relations of words, and keyword-based IR techniques are insufficient to describe such relations precisely.

This paper proposes a practical HPSG parser for English, **Enju**, an intelligent search engine for the accurate retrieval of relational concepts from MEDLINE, **MEDIE**, and a GUI-based efficient MEDLINE search tool, **Info-PubMed**.

	F-Score	
	GENIA treebank	Penn Treebank
HPSG-PTB	85.10%	87.16%
HPSG-GENIA	86.87%	86.81%

Table 1: Performance for Penn Treebank and the GENIA corpus

2 Enju: An English HPSG Parser

We developed an English HPSG parser, Enju [1] (Miyao and Tsujii, 2005; Hara et al., 2005; Ninomiya et al., 2005). Table 1 shows the performance. The F-score in the table was accuracy of the predicate-argument relations output by the parser. A predicate-argument relation is defined as a tuple $\langle \sigma, w_h, a, w_a \rangle$, where σ is the predicate type (e.g., adjective, intransitive verb), w_h is the head word of the predicate, a is the argument label (**MOD, ARG1, ..., ARG4**), and w_a is the head word of the argument. Precision/recall is the ratio of tuples correctly identified by the parser. The lexicon of the grammar was extracted from Sections 02-21 of Penn Treebank (39,832 sentences). In the table, 'HPSG-PTB' means that the statistical model was trained on Penn Treebank. 'HPSG-GENIA' means that the statistical model was trained on both Penn Treebank and GENIA treebank as described in (Hara et al., 2005). The GENIA treebank (Tateisi et al., 2005) consists of 500 abstracts (4,446 sentences) extracted from MEDLINE.

Figure 1 shows a part of the parse tree and fea-

[*] Current Affiliation:
[†] School of Informatics, University of Manchester
[‡] Knowledge Research Center, Fujitsu Laboratories LTD.
[§] Faculty of Informatics, Kogakuin University
[¶] Information Technology Center, University of Tokyo

[1] http://www-tsujii.is.s.u-tokyo.ac.jp/enju/

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 17–20,
Sydney, July 2006.

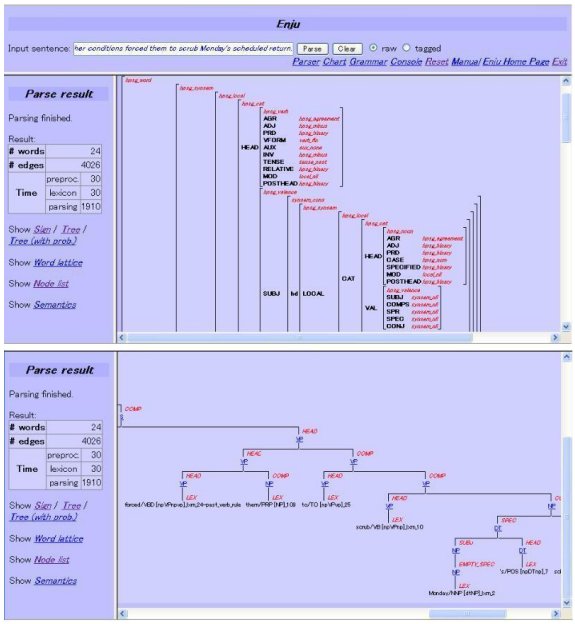


Figure 1: Snapshot of Enju

ture structure for the sentence “NASA officials vowed to land Discovery early Tuesday at one of three locations after weather conditions forced them to scrub Monday’s scheduled return.”

3 MEDIE: a search engine for MEDLINE

Figure 2 shows the top page of the MEDIE. MEDIE is an intelligent search engine for the accurate retrieval of relational concepts from MEDLINE [2] (Miyao et al., 2006). Prior to retrieval, all sentences are annotated with predicate argument structures and ontological identifiers by applying Enju and a term recognizer.

3.1 Automatically Annotated Corpus

First, we applied a POS analyzer and then Enju. The POS analyzer and HPSG parser are trained by using the GENIA corpus (Tsuruoka et al., 2005; Hara et al., 2005), which comprises around 2,000 MEDLINE abstracts annotated with POS and Penn Treebank style syntactic parse trees (Tateisi et al., 2005). The HPSG parser generates parse trees in a stand-off format that can be converted to XML by combining it with the original text.

We also annotated technical terms of genes and diseases in our developed corpus. Technical terms are annotated simply by exact matching of dictionary entries and the terms separated by space, tab, period, comma, hat, colon, semi-colon, brackets, square brackets and slash in MEDLINE.

The entire dictionary was generated by applying the automatic generation method of name variations (Tsuruoka and Tsujii, 2004) to the GENA dictionary for the gene names (Koike and Takagi, 2004) and the UMLS (Unified Medical Language System) meta-thesaurus for the disease names (Lindberg et al., 1993). It was generated by applying the name-variation generation method, and we obtained 4,467,855 entries of a gene and disease dictionary.

3.2 Functions of MEDIE

MEDIE provides three types of search, **semantic search**, **keyword search**, **GCL search**. GCL search provides us the most fundamental and powerful functions in which users can specify the boolean relations, linear order relation and structural relations with variables. Trained users can enjoy all functions in MEDIE by the GCL search, but it is not easy for general users to write appropriate queries for the parsed corpus. The semantic search enables us to specify an event verb with its subject and object easily. MEDIE automatically generates the GCL query from the semantic query, and runs the GCL search. Figure 3 shows the output of semantic search for the query ‘What disease does dystrophin cause?’. This example will give us the most intuitive understandings of the proximal and structural retrieval with a richly annotated parsed corpus. MEDIE retrieves sentences which include event verbs of ‘*cause*’ and noun ‘*dystrophin*’ such that ‘dystrophin’ is the subject of the event verbs. The event verb and its subject and object are highlighted with designated colors. As seen in the figure, small sentences in relative clauses, passive forms or coordination are retrieved. As the objects of the event verbs are highlighted, we can easily see what disease dystrophin caused. As the target corpus is already annotated with diseases entities, MEDIE can efficiently retrieve the disease expressions.

4 Info-PubMed: a GUI-based MEDLINE search tool

Info-PubMed is a MEDLINE search tool with GUI, helping users to find information about biomedical entities such as *genes*, *proteins*, and

[2] http://www-tsujii.is.s.u-tokyo.ac.jp/medie/

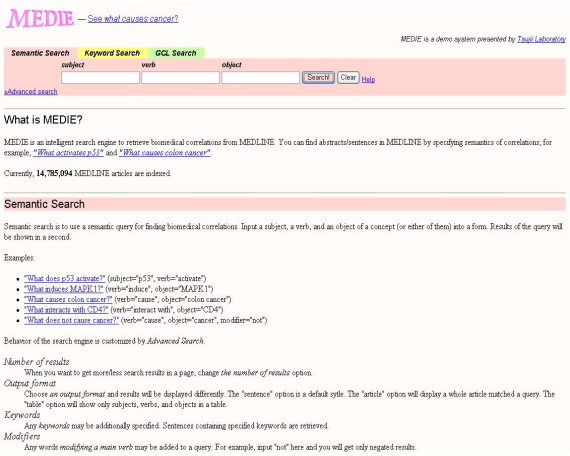


Figure 2: Snapshot of MEDIE: top page‘

Figure 3: Snapshot of MEDIE: ‘What disease does dystrophin cause?’

the *interactions* between them [3].

Info-PubMed provides information from MEDLINE on protein-protein interactions. Given the name of a *gene* or *protein*, it shows a list of the names of other *genes/proteins* which co-occur in sentences from MEDLINE, along with the frequency of co-occurrence.

Co-occurrence of two *proteins/genes* in the same sentence does not always imply that they interact. For more accurate extraction of sentences that indicate interactions, it is necessary to identify relations between the two substances. We adopted PASs derived by Enju and constructed extraction patterns on specific verbs and their arguments based on the derived PASs (Yakusiji, 2006).

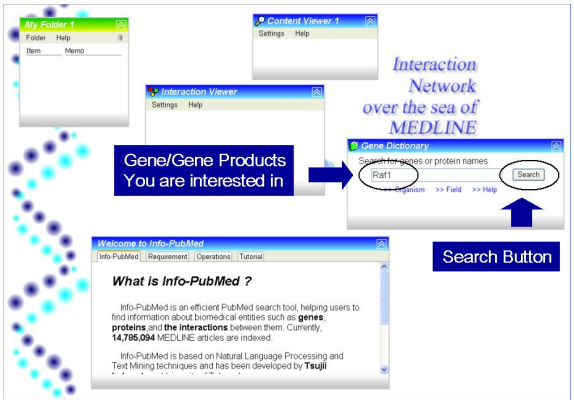


Figure 4: Snapshot of Info-PubMed (1)

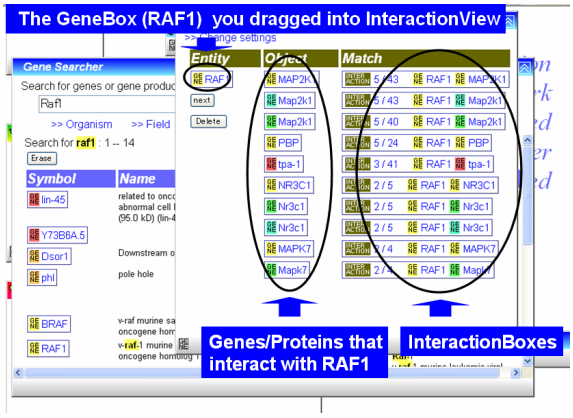


Figure 5: Snapshot of Info-PubMed (2)

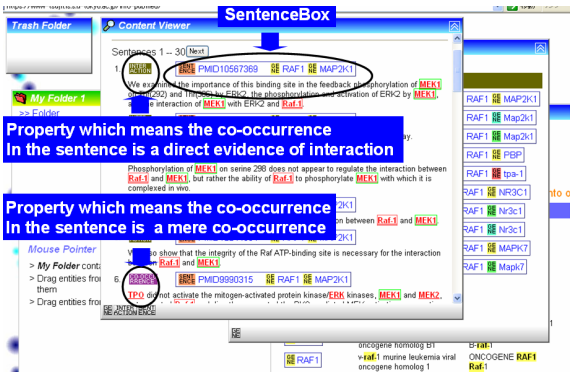


Figure 6: Snapshot of Info-PubMed (3)

4.1 Functions of Info-PubMed

In the ‘Gene Searcher’ window, enter the name of a gene or protein that you are interested in. For example, if you are interested in Raf1, type “raf1” in the ‘Gene Searcher’ (Figure 4). You will see a list of genes whose description in our dictionary contains “raf1” (Figure 5). Then, drag

[3] http://www-tsujii.is.s.u-tokyo.ac.jp/info-pubmed/

one of the GeneBoxes from the 'Gene Searcher' to the 'Interaction Viewer.' You will see a list of genes/proteins which co-occur in the same sentences, along with co-occurrence frequency. The GeneBox in the leftmost column is the one you have moved to 'Interaction Viewer.' The GeneBoxes in the second column correspond to gene/proteins which co-occur in the same sentences, followed by the boxes in the third column, InteractionBoxes.

Drag an InteractionBox to 'ContentViewer' to see the content of the box (Figure 6). An InteractionBox is a set of SentenceBoxes. A SentenceBox corresponds to a sentence in MEDLINE in which the two gene/proteins co-occur. A SentenceBox indicates whether the co-occurrence in the sentence is direct evidence of interaction or not. If it is judged as direct evidence of interaction, it is indicated as Interaction. Otherwise, it is indicated as Co-occurrence.

5 Conclusion

We presented an English HPSG parser, **Enju**, a search engine for relational concepts from MEDLINE, **MEDIE**, and a GUI-based MEDLINE search tool, **Info-PubMed**.

MEDIE and Info-PubMed demonstrate how the results of deep parsing can be used for intelligent text mining and semantic information retrieval in the biomedical domain.

6 Acknowledgment

This work was partially supported by Grant-in-Aid for Scientific Research on Priority Areas "Systems Genomics" (MEXT, Japan) and Solution-Oriented Research for Science and Technology (JST, Japan).

References

C. Blaschke and A. Valencia. 2002. The frame-based module of the SUISEKI information extraction system. *IEEE Intelligent Systems*, 17(2):14–20.

Y. Hao, X. Zhu, M. Huang, and M. Li. 2005. Discovering patterns to extract protein-protein interactions from the literature: Part II. *Bioinformatics*, 21(15):3294–3300.

H.-W. Chun, Y. Tsuruoka, J.-D. Kim, R. Shiba, N. Nagata, T. Hishiki, and J. Tsujii. 2006. Extraction of gene-disease relations from MedLine using domain dictionaries and machine learning. In *Proc. PSB 2006*, pages 4–15.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Yusuke Miyao and Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proc. of ACL'05*, pages 83–90.

Tadayoshi Hara, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In *Proc. of IJCNLP 2005*.

Takashi Ninomiya, Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Efficacy of beam thresholding, unification filtering and hybrid parsing in probabilistic HPSG parsing. In *Proc. of IWPT 2005*, pages 103–114.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax Annotation for the GENIA corpus. In *Proc. of the IJCNLP 2005, Companion volume*, pp. 222–227.

Yusuke Miyao, Tomoko Ohta, Katsuya Masuda, Yoshimasa Tsuruoka, Kazuhiro Yoshida, Takashi Ninomiya and Jun'ichi Tsujii. 2006. Semantic Retrieval for the Accurate Identification of Relational Concepts in Massive Textbases. In *Proc. of ACL '06*, to appear.

Yoshimasa Tsuruoka, Yuka Tateisi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii. 2005. Part-of-speech tagger for biomedical text. In *Proc. of the 10th Panhellenic Conference on Informatics*.

Y. Tsuruoka and J. Tsujii. 2004. Improving the performance of dictionary-based approaches in protein name recognition. *Journal of Biomedical Informatics*, 37(6):461–470.

Asako Koike and Toshihisa Takagi. 2004. Gene/protein/family name recognition in biomedical literature. In *Proc. of HLT-NAACL 2004 Workshop: Biolink 2004*, pages 9–16.

D.A. Lindberg, B.L. Humphreys, and A.T. McCray. 1993. The unified medical language system. *Methods in Inf. Med.*, 32(4):281–291.

Akane Yakushiji. 2006. Relation Information Extraction Using Deep Syntactic Analysis. *Ph.D. Thesis*, University of Tokyo.

MIMA Search: A Structuring Knowledge System towards Innovation for Engineering Education

Hideki Mima
School of Engineering
University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033, Japan
mima@t-adm.t.u-tokyo.ac.jp

Abstract

The main aim of the MIMA (Mining Information for Management and Acquisition) Search System is to achieve 'structuring knowledge' to accelerate knowledge exploitation in the domains of science and technology. This system integrates natural language processing including ontology development, information retrieval, visualization, and database technology. The 'structuring knowledge' that we define indicates 1) knowledge storage, 2) (hierarchical) classification of knowledge, 3) analysis of knowledge, 4) visualization of knowledge. We aim at integrating different types of databases (papers and patents, technologies and innovations) and knowledge domains, and simultaneously retrieving different types of knowledge. Applications for the several targets such as syllabus structuring will also be mentioned.

1 Introduction

The growing number of electronically available knowledge sources (KSs) emphasizes the importance of developing flexible and efficient tools for automatic knowledge acquisition and structuring in terms of knowledge integration. Different text and literature mining techniques have been developed recently in order to facilitate efficient discovery of knowledge contained in large textual collections. The main goal of literature mining is to retrieve knowledge that is "buried" in a text and to present the distilled knowledge to users in a concise form. Its advantage, compared to "manual" knowledge discovery, is based on the assumption that automatic methods are able to process an enormous amount of text. It is doubtful that any researcher could process such a huge amount of information, especially if the knowledge spans across domains. For these reasons, literature mining aims at helping scientists in collecting, maintaining, interpreting and curating information.

In this paper, we introduce a knowledge structuring system (KSS) we designed, in which terminology-driven knowledge acquisition (KA), knowledge retrieval (KR) and knowledge visualization (KV) are combined using automatic term recognition, automatic term clustering and terminology-based similarity calculation is explained. The system incorporates our proposed automatic term recognition / clustering and a visualization of retrieved knowledge based on the terminology, which allow users to access KSs visually though sophisticated GUIs.

2 Overview of the system

The main purpose of the knowledge structuring system is 1) accumulating knowledge in order to develop huge knowledge bases, 2) exploiting the accumulated knowledge efficiently. Our approach to structuring knowledge is based on:

- automatic term recognition (ATR)
- automatic term clustering (ATC) as an ontology[1] development
- ontology-based similarity calculation
- visualization of relationships among documents (KSs)

One of our definitions to structuring knowledge is discovery of relevance between documents (KSs) and its visualization. In order to achieve real time processing for structuring knowledge, we adopt terminology / ontology-based similarity calculation, because knowledge can also be represented as textual documents or passages (e.g. sentences, subsections) which are efficiently characterized by sets of specialized (technical) terms. Further details of our visualization scheme will be mentioned in Section 4.

[1] Although, definition of ontology is domain-specific, our definition of ontology is the collection and classification of (technical) terms to recognize their semantic relevance.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 21–24,
Sydney, July 2006.

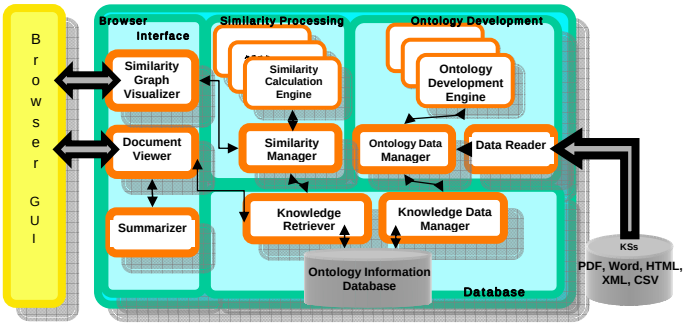


Figure 1: The system architecture

The system architecture is modular, and it integrates the following components (Figure 1):

- *Ontology Development Engine(s) (ODE)* – components that carry out the automatic ontology development which includes recognition and structuring of domain terminology;
- *Knowledge Data Manager (KDM)* – stores index of KSs and ontology in a ontology information database (OID) and provides the corresponding interface;
- *Knowledge Retriever (KR)* – retrieves KSs from TID and calculates similarities between keywords and KSs. Currently, we adopt tf*idf based similarity calculation;
- *Similarity Calculation Engine(s) (SCE)* – calculate similarities between KSs provided from KR component using ontology developed by ODE in order to show semantic similarities between each KSs. We adopt Vector Space Model (VSM) based similarity calculation and use terms as features of VSM. Semantic clusters of KSs are also provided.
- *Graph Visualizer* – visualizes knowledge structures based on graph expression in which relevance links between provided keywords and KSs, and relevance links between the KSs themselves can be shown.

3 Terminological processing as an ontology development

The lack of clear naming standards in a domain (e.g. biomedicine) makes ATR a non-trivial problem (Fukuda et al., 1998). Also, it typically gives rise to many-to-many relationships between terms and concepts. In practice, two problems stem from this fact: 1) there are terms that have multiple meanings (*term ambiguity*), and, conversely, 2) there are terms that refer to the same concept (*term variation*). Generally, term ambiguity has negative effects on IE precision, while term variation decreases IE recall. These problems show the difficulty of using simple keyword-based IE techniques. Obviously, more sophisticated techniques, identifying groups of different terms referring to the same (or similar) concept(s), and, therefore, could benefit from relying on efficient and consistent ATR/ATC and term variation management methods are required. These methods are also important for organising domain specific knowledge, as terms should not be treated isolated from other terms. They should rather be related to one another so that the relations existing between the corresponding concepts are at least partly reflected in a terminology.

3.1 Term recognition

The ATR method used in the system is based on the *C / NC-value* methods (Mima et al., 2001; Mima and Ananiadou, 2001). The *C-value* method recognizes terms by combining linguistic knowledge and statistical analysis. The method extracts multi-word terms[2] and is not limited to a specific class of concepts. It is implemented as a two-step procedure. In the first step, term candidates are extracted by using a set of linguistic filters which describe general term formation patterns. In the second step, the term candidates are assigned termhood scores (referred to as *C-values*) according to a statistical measure. The measure amalgamates four numerical corpus-based characteristics of a candidate term, namely the frequency of occurrence, the frequency of occurrence as a substring of other candidate terms, the number of candidate terms containing the given candidate term as a substring, and the number of words contained in the candidate term.

The *NC-value method* further improves the C-value results by taking into account the context of candidate terms. The relevant context words are extracted and assigned weights based on how frequently they appear with top-ranked term candidates extracted by the *C-value* method. Subsequently, context factors are assigned to candidate terms according to their co-occurrence with top-ranked context words. Finally, new termhood estimations, referred to as *NC-values*, are calculated as a linear combination of the *C-values* and context factors for the respective terms. Evaluation of the *C/NC-methods* (Mima and Ananiadou, 2001) has shown that contextual information improves term distribution in the extracted list by placing real terms closer to the top of the list.

[2] More than 85% of domain-specific terms are multi-word terms (Mima and Ananiadou, 2001).

3.2 Term variation management

Term variation and ambiguity are causing problems not only for ATR but for human experts as well. Several methods for term variation management have been developed. For example, the BLAST system Krauthammer et al., 2000) used approximate text string matching techniques and dictionaries to recognize spelling variations in gene and protein names. FASTR (Jacquemin, 2001) handles morphological and syntactic variations by means of meta-rules used to describe term normalization, while semantic variants are handled via WordNet.

The basic *C-value* method has been enhanced by term variation management (Mima and Ananiadou, 2001). We consider a variety of sources from which term variation problems originate. In particular, we deal with orthographical, morphological, syntactic, lexico-semantic and pragmatic phenomena. Our approach to term variation management is based on term normalization as an integral part of the ATR process. Term variants (i.e. synonymous terms) are dealt with in the initial phase of ATR when term candidates are singled out, as opposed to other approaches (e.g. FASTR handles variants subsequently by applying transformation rules to extracted terms). Each term variant is normalized (see table 1 as an example) and term variants having the same normalized form are then grouped into classes in order to link each term candidate to all of its variants. This way, a list of normalized term candidate classes, rather than a list of single terms is statistically processed. The termhood is then calculated for a whole class of term variants, not for each term variant separately.

Table 1: Automatic term normalization

Term variants	Normalised term
human cancers cancer in humans human's cancer human carcinoma	→ human cancer

3.3 Term clustering

Beside term recognition, term clustering is an indispensable component of the literature mining process. Since terminological opacity and polysemy are very common in molecular biology and biomedicine, term clustering is essential for the semantic integration of terms, the construction of domain ontologies and semantic tagging.

ATC in our system is performed using a hierarchical clustering method in which clusters are merged based on average mutual information measuring how strongly terms are related to one another (Ushioda, 1996). Terms automatically recognized by the NC-value method and their co-occurrences are used as input, and a dendrogram of terms is produced as output. Parallel symmetric processing is used for high-speed clustering. The calculated term cluster information is encoded and used for calculating semantic similarities in SCE component. More precisely, the similarity between two individual terms is determined according to their position in a dendrogram. Also a commonality measure is defined as the number of shared ancestors between two terms in the dendrogram, and a positional measure as a sum of their distances from the root. Similarity between two terms corresponds to a ratio between commonality and positional measure.

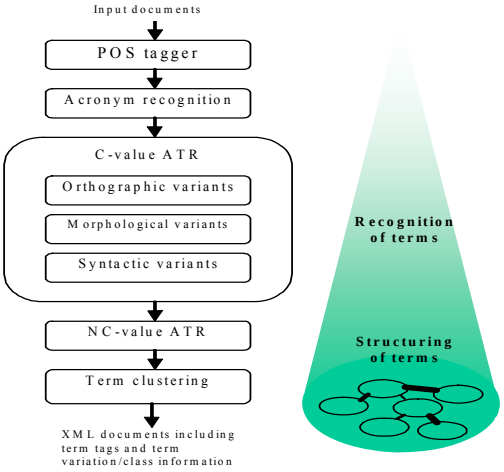


Figure 2: Ontology development

Further details of the methods and their evaluations can be referred in (Mima et al., 2001; Mima and Ananiadou, 2001).

4 Structuring knowledge

Structuring knowledge can be regarded as a broader approach to IE/KA. IE and KA in our system are implemented through the integration of ATR, ATC, and ontology-based semantic similarity calculation. Graph-based visualization for globally structuring knowledge is also provided to facilitate KR and KA from documents. Additionally, the system supports combining different databases (papers and patents, technologies and innovations) and retrieves different types of knowledge simultaneously and crossly. This feature can accelerate knowledge discovery by combining existing knowledge. For example, discovering new knowledge on industrial innovation by structuring knowledge of trendy scientific paper database and past industrial innovation report database can be expected. Figure 3 shows an example of visualization of knowledge structures in the

domain of engineering. In order to structure knowledge, the system draws a graph in which nodes indicate relevant KSs to keywords given and each links between KSs indicates semantic similarities dynamically calculated using ontology information developed by our ATR / ATC components.

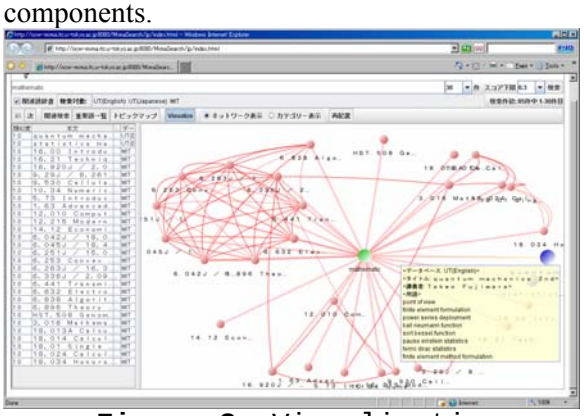


Figure 3: Visualization

5 Conclusion

In this paper, we presented a system for structuring knowledge over large KSs. The system is a terminology-based integrated KA system, in which we have integrated ATR, ATC, IR, similarity calculation, and visualization for structuring knowledge. It allows users to search and combine information from various sources. KA within the system is terminology-driven, with terminology information provided automatically. Similarity based knowledge retrieval is implemented through various semantic similarity calculations, which, in combination with hierarchical, ontology- based matching, offers powerful means for KA through visualization-based literature mining.

We have applied the system to syllabus retrieval for The University of Tokyo`s Open Course Ware (UT-OCW)[3] site and syllabus structuring (SS) site[4] for school / department of engineering at University of Tokyo, and they are both available in public over the Internet. The UT-OCW's MIMA Search system is designed to search the syllabuses of courses posted on the UT-OCW site and the Massachusetts Institute of Technology's OCW site (MIT-OCW). Also, the SS site's MIMA Search is designed to search the syllabuses of lectures from more than 1,600 lectures in school / department of engineering at University of Tokyo. Both systems show search results in terms of relations among the syllabuses as a structural graphic (figure 3). Based on the automatically extracted terms from the syllabuses and similarities calculated using those terms, MIMA Search displays the search results in a network format, using dots and lines. Namely, MIMA Search extracts the contents from the listed syllabuses, rearrange these syllabuses according to semantic relations of the contents and display the results graphically, whereas conventional search engines simply list the syllabuses that are related to the keywords. Thanks to this process, we believe users are able to search for key information and obtain results in minimal time. In graphic displays, as already mentioned, the searched syllabuses are shown in a structural graphic with dots and lines. The stronger the semantic relations of the syllabuses, the closer they are placed on the graphic. This structure will help users find a group of courses / lectures that are closely related in contents, or take courses / lectures in a logical order, for example, beginning with fundamental mathematics and going on to applied mathematics. Furthermore, because of the structural graphic display, users will be able to instinctively find the relations among syllabuses of other universities.

Currently, we obtain more than 2,000 hits per day in average from all over the world, and have provided more then 50,000 page views during last three months. On the other hand, we are in a process of system evaluation using more than 40 students to evaluate usability as a next generation information retrieval.

The other experiments we conducted also show that the system's knowledge structuring scheme is an efficient methodology to facilitate KA and new knowledge discovery in the field of genome and nano-technology (Mima et al., 2001).

[3] http://ocw.u-tokyo.ac.jp/.
[4] http://ciee.t.u-tokyo.ac.jp/.

References

K. Fukuda, T. Tsunoda, A. Tamura, T. Takagi, 1998. Toward information extraction: identifying protein names from biological papers, Proc. of PSB-98, Hawaii, pp. 3:705-716.

H. Mima, S. Ananiadou, G. Nenadic, 2001. ATRACT workbench: an automatic term recognition and clustering of terms, in: V. Matoušek, P. Mautner, R. Mouček, K. Tašer (Eds.) Text, Speech and Dialogue, LNAI 2166, Springer Verlag, pp. 126-133.

H. Mima, S. Ananiadou, 2001. An application and evaluation of the C/NC-value approach for the automatic term recognition of multi-word units in Japanese, Int. J. on Terminology 6/2, pp. 175-194.

M. Krauthammer, A. Rzhetsky, P. Morozov, C. Friedman, 2000. Using BLAST for identifying gene and protein names in journal articles, in: Gene 259, pp. 245-252.

C. Jacquemin, 2001. Spotting and discovering terms through NLP, MIT Press, Cambridge MA, p. 378.

A. Ushioda, 1996. Hierarchical clustering of words, Proc. of COLING '96, Copenhagen, Denmark, pp. 1159-1162.

FERRET: Interactive Question-Answering for Real-World Environments

Andrew Hickl, Patrick Wang, John Lehmann, and Sanda Harabagiu
Language Computer Corporation
1701 North Collins Boulevard
Richardson, Texas 75080 USA
ferret@languagecomputer.com

Abstract

This paper describes FERRET, an interactive question-answering (Q/A) system designed to address the challenges of integrating automatic Q/A applications into real-world environments. FERRET utilizes a novel approach to Q/A – known as *predictive questioning* – which attempts to identify the questions (and answers) that users need by analyzing how a user interacts with a system while gathering information related to a particular scenario.

1 Introduction

As the accuracy of today's best factoid question-answering (Q/A) systems (Harabagiu et al., 2005; Sun et al., 2005) approaches 70%, research has begun to address the challenges of integrating automatic Q/A systems into real-world environments. A new class of applications – known as interactive Q/A systems – are now being developed which allow users to ask questions in the context of extended dialogues in order to gather information related to any number of complex scenarios. In this paper, we describe our interactive Q/A system – known as FERRET – which uses an approach based on *predictive questioning* in order to meet the changing information needs of users over the course of a Q/A dialogue.

Answering questions in an interactive setting poses three new types of challenges for traditional Q/A systems. First, since current Q/A systems are designed to answer single questions in isolation, interactive Q/A systems must look for ways to foster interaction with a user throughout all phases of the research process. Unlike traditional Q/A applications, interactive Q/A systems must do more than cooperatively answer a user's single question. Instead, in order to keep a user collaborating with the system, interactive Q/A systems need to provide access to new types of information that are somehow relevant to the user's stated – and unstated – information needs.

Second, we have found that users of Q/A systems in real-world settings often ask questions that are much more complex than the types of factoid questions that have been evaluated in the annual Text Retrieval Conference (TREC) evaluations. When faced with a limited period of time to gather information, even experienced users of Q/A may find it difficult to translate their information needs into the simpler types of questions that Q/A systems can answer. In order to provide effective answers to these questions, interactive question-answering systems need to include *question decomposition* techniques that can break down complex questions into the types of simpler factoid-like questions that traditional Q/A systems were designed to answer.

Finally, interactive Q/A systems must be sensitive not only to the content of a user's question – but also to the context that it is asked in. Like other types of task-oriented dialogue systems, interactive Q/A systems need to model both what a user knows – and what a user wants to know – over the course of a Q/A dialogue: systems that fail to represent a user's knowledge base run the risk of returning redundant information, while systems that do not model a user's intentions can end up returning irrelevant information.

In the rest of this paper, we discuss how the FERRET interactive Q/A system currently addresses the first two of these three challenges.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 25–28,
Sydney, July 2006.

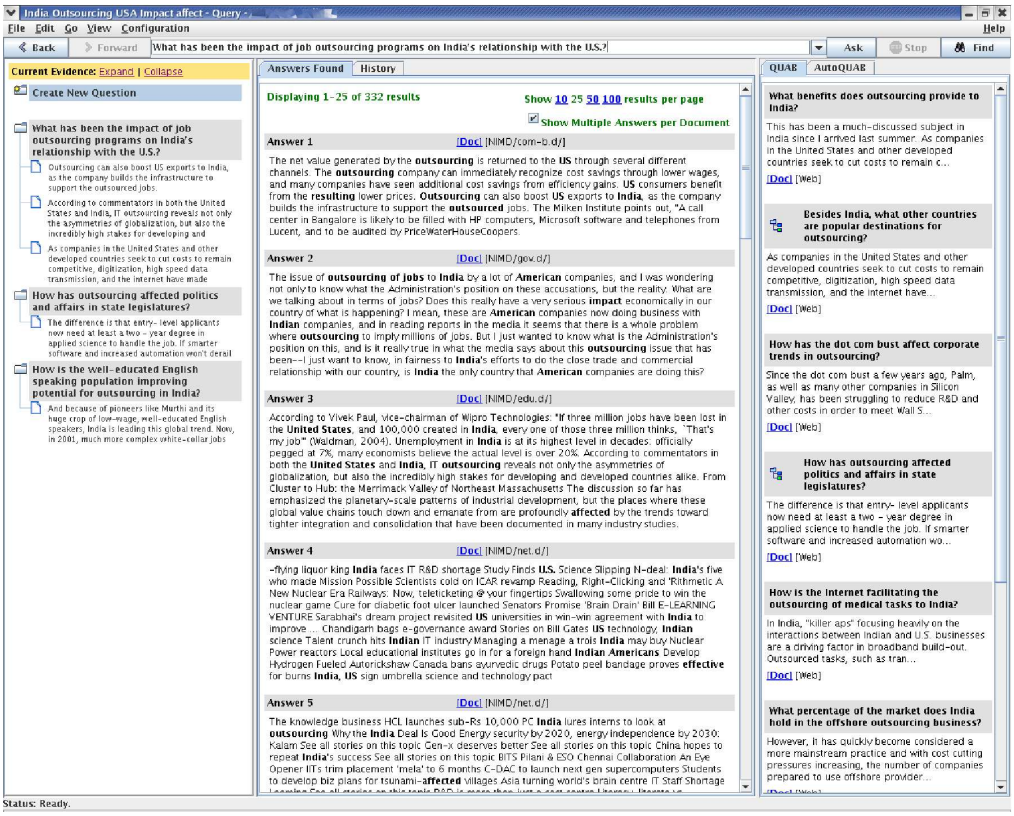


Figure 1: The FERRET Interactive Q/A System

2 The FERRET Interactive Question-Answering System

This section provides a basic overview of the functionality provided by the FERRET interactive Q/A system. [1]

FERRET returns three types of information in response to a user's query. First, FERRET utilizes an automatic Q/A system to find answers to users' questions in a document collection. In order to provide users with the timely results that they expect from information gathering applications (such as Internet search engines), every effort was made to reduce the time FERRET takes to extract answers from text. (In the current version of the system, answers are returned on average in 12.78 seconds. [2])

In addition to answers, FERRET also provides information in the form of two different types of *predictive question-answer pairs* (or QUABs). With FERRET, users can select from QUABs that were either generated automatically from the set of documents returned by the Q/A system or that were selected from a large database of more than 10,000 question-answer pairs created offline by human annotators. In the current version of FERRET, the top 10 automatically-generated and hand-crafted QUABs that are most judged relevant to the user's original question are returned to the user as potential continuations of the dialogue. Each set of QUABs is presented in a separate pane found to the right of the answers returned by the Q/A system; QUABs are ranked in order of relevance to the user's original query.

Figure 1 provides a screen shot of FERRET's interface. Q/A answers are presented in the center pane of the FERRET browser, while QUAB question-answer pairs are presented in two separate tabs found in the rightmost pane of the browser. FERRET's leftmost pane includes a "drag-and-drop" clipboard which facilitates note-taking and annotation over the course of an interactive Q/A dialogue.

3 Predictive Question-Answering

First introduced in (Harabagiu et al., 2005b), a *predictive questioning* approach to automatic

[1] For more details on FERRET's question-answering capabilities, the reader is invited to consult (Harabagiu et al., 2005a); for more information on FERRET's predictive question generation component, please see (Harabagiu et al., 2005b).

[2] This test was run on a machine with a Pentium 4 3.0 GHz processor with 2 GB of RAM.

question-answering assumes that Q/A systems can use the set of documents relevant to a user's query in order to generate sets of questions – known as *predictive questions* – that anticipate a user's information needs. Under this approach, topic representations like those introduced in (Lin and Hovy, 2000) and (Harabagiu, 2004) are used to identify a set of text passages that are relevant to a user's domain of interest. Topic-relevant passages are then semantically parsed (using a PropBank-style semantic parser) and submitted to a question generation module, which uses a set of syntactic rewrite rules in order to create natural language questions from the original passage.

Generated questions are then assembled into question-answer pairs – known as QUABs – with the original passage serving as the question's "answer", and are then returned to the user. For example, two of the predictive question-answer pairs generated from the documents returned for question Q_0, "*What has been the impact of job outsourcing programs on India's relationship with the U.S.?*", are presented in Table 1.

Q_0	What has been the impact of job outsourcing programs on India's relationship with the U.S.?
PQ_1	How could India respond to U.S. efforts to limit job outsourcing?
A_1	U.S. officials have countered that the best way for India to counter U.S. efforts to limit job outsourcing is to further liberalize its markets.
PQ_2	What benefits does outsourcing provide to India?
A_2	India's prowess in outsourcing is no longer the only reason why outsourcing to India is an attractive option. The difference lies in the scalability of major Indian vendors, their strong focus on quality and their experience delivering a wide range of services", says John Blanco, senior vice president at Cablevision Systems Corp. in Bethpage, N.Y.
PQ_2	Besides India, what other countries are popular destinations for outsourcing?
A_2	A number of countries are now beginning to position themselves as outsourcing centers including China, Russia, Malaysia, the Philippines, South Africa and several countries in Eastern Europe.

Table 1: Predictive Question-Answer Pairs

While neither PQ_1 nor PQ_2 provide users with an exact answer to the original question Q_0, both QUABs can be seen as providing users information which is complementary to acquiring information on the topic of *job outsourcing*: PQ_1 provides details on how India could respond to anti-outsourcing legislation, while PQ_2 talks about other countries that are likely targets for outsourcing.

We believe that QUABs can play an important role in fostering extended dialogue-like interactions with users. We have observed that the incorporation of predictive-question answer pairs into an interactive question-answering system like FERRET can promote dialogue-like interactions between users and the system. When presented with a set of QUAB questions, users typically selected a coherent set of follow-on questions which served to elaborate or clarify their initial question. The dialogue fragment in Table 2 provides an example of the kinds of dialogues that users can generate by interacting with the predictive questions that FERRET generates.

UserQ_1:	What has been the impact of job outsourcing programs on India's relationship with the U.S.?
QUAB_1:	How could India respond to U.S. efforts to limit job outsourcing?
QUAB_2:	Besides India, what other countries are popular destinations for outsourcing?
UserQ_2:	What industries are outsourcing jobs to India?
QUAB_3:	Which U.S. technology companies have opened customer service departments in India?
QUAB_4:	Will Dell follow through on outsourcing technical support jobs to India?
QUAB_5:	Why do U.S. companies find India an attractive destination for outsourcing?
UserQ_3:	What anti-outsourcing legislation has been considered in the U.S.?
QUAB_6:	Which Indiana legislator introduced a bill that would make it illegal to outsource Indiana jobs?
QUAB_7:	What U.S. Senators have come out against anti-outsourcing legislation?

Table 2: Dialogue Fragment

In experiments with human users of FERRET, we have found that QUAB pairs enhanced the quality of information retrieved that users were able to retrieve during a dialogue with the system. [3] In 100 user dialogues with FERRET, users clicked hyperlinks associated with QUAB pairs 56.7% of the time, despite the fact the system returned (on average) approximately 20 times more answers than QUAB pairs. Users also derived value from information contained in QUAB pairs: reports written by users who had access to QUABs while gathering information were judged to be significantly ($p < 0.05$) better than those reports written by users who only had access to FERRET's Q/A system alone.

4 Answering Complex Questions

As was mentioned in Section 2, FERRET uses a special dialogue-optimized version of an automatic question-answering system in order to find high-precision answers to users' questions in a document collection.

During a Q/A dialogue, users of interactive Q/A systems frequently ask complex questions that must be decomposed syntactically and semantically before they can be answered using traditional Q/A techniques. Complex questions submitted to

[3] For details of user experiments with FERRET, please see (Harabagiu et al., 2005b).

FERRET are first subject to a set of syntactic decomposition heuristics which seek to extract each overtly-mentioned subquestion from the original question. Under this approach, questions featuring coordinated question stems, entities, verb phrases, or clauses are split into their separate conjuncts; answers to each syntactically decomposed question are presented separately to the user. Table 3 provides an example of syntactic decomposition performed in FERRET.

CQ_1	What industries have been outsourcing or offshoring jobs to India or Malaysia?
QD_1	What industries have been outsourcing jobs to India?
QD_2	What industries have been offshoring jobs to India?
QD_3	What industries have been outsourcing jobs to Malaysia?
QD_4	What industries have been offshoring jobs to Malaysia?

Table 3: Syntactic Decomposition

FERRET also performs semantic decomposition of complex questions using techniques first outlined in (Harabagiu et al., 2006). Under this approach, three types of semantic and pragmatic information are identified in complex questions: (1) information associated with a complex question's expected answer type, (2) semantic dependencies derived from predicate-argument structures discovered in the question, and (3) and topic information derived from documents retrieved using the keywords contained the question. Examples of the types of automatic semantic decomposition that is performed in FERRET is presented in Table 4.

CQ_2	What has been the impact of job outsourcing programs on India's relationship with the U.S.?
QD_5	What is meant by India's relationship with the U.S.?
QD_6	What outsourcing programs involve India and the U.S.?
QD_7	Who has started outsourcing programs for India and the U.S.?
QD_8	What statements were made regarding outsourcing on India's relationship with the U.S.?

Table 4: Semantic Question Decomposition

Complex questions are decomposed by a procedure that operates on a Markov chain, by following a random walk on a bipartite graph of question decompositions and relations relevant to the topic of the question. Unlike with syntactic decomposition, FERRET combines answers from semantically decomposed question automatically and presents users with a single set of answers that represents the contributions of each question. Users are notified that semantic decomposition has occurred, however; decomposed questions are displayed to the user upon request.

In addition to techniques for answering complex questions, FERRET's Q/A system improves performance for a variety of question types by employing separate question processing strategies in order to provide answers to four different types of questions, including factoid questions, list questions, relationship questions, and definition questions.

5 Conclusions

We created FERRET as part of a larger effort designed to address the challenges of integrating automatic question-answering systems into real-world research environments. We have focused on two components that have been implemented into the latest version of FERRET: (1) predictive questioning, which enables systems to provide users with question-answer pairs that may anticipate their information needs, and (2) question decomposition, which serves to break down complex questions into sets of conceptually-simpler questions that Q/A systems can answer successfully.

6 Acknowledgments

This material is based upon work funded in whole or in part by the U.S. Government and any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

References

S. Harabagiu, D. Moldovan, C. Clark, M. Bowden, A. Hickl, and P. Wang. 2005a. Employing Two Question Answering Systems in TREC 2005. In *Proceedings of the Fourteenth Text REtrieval Conference*.

Sanda Harabagiu, Andrew Hickl, John Lehmann, and Dan Moldovan. 2005b. Experiments with Interactive Question-Answering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.

Sanda Harabagiu, Finley Lacatusu, and Andrew Hickl. 2006. Answering complex questions with random walk models. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, WA.

Sanda Harabagiu. 2004. Incremental Topic Representations. In *Proceedings of the 20th COLING Conference*, Geneva, Switzerland.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th COLING Conference*, Saarbrücken, Germany.

R. Sun, J. Jiang, Y. F. Tan, H. Cui, T.-S. Chua, and M.-Y. Kan. 2005. Using Syntactic and Semantic Relation Analysis in Question Answering. In *Proceedings of The Fourteenth Text REtrieval Conference (TREC 2005)*.

K-QARD: A Practical Korean Question Answering Framework for Restricted Domain

Young-In Song, HooJung Chung, Kyoung-Soo Han, JooYoung Lee, Hae-Chang Rim
Dept. of Computer Science & Engineering
Korea University
Seongbuk-gu, Seoul 136-701, Korea
{song, hjchung, kshan, jylee rim}@nlp.korea.ac.kr

Jae-Won Lee
Computing Lab.
Samsung Advanced Institute of Technology
Nongseo-ri, Giheung-eup,
Yongin-si, Gyeonggi-do 449-712, Korea
jwonlee@samsung.com

Abstract

We present a Korean question answering framework for restricted domains, called K-QARD. K-QARD is developed to achieve domain portability and robustness, and the framework is successfully applied to build question answering systems for several domains.

1 Introduction

K-QARD is a framework for implementing a fully automated question answering system including the Web information extraction (IE). The goal of the framework is to provide a practical environment for the restricted domain question answering (QA) system with the following requirements:

- **Domain portability**: Domain adaptation of QA systems based on the framework should be possible with minimum human efforts.
- **Robustness**: The framework has to provide methodologies to ensure robust performance for various expressions of a question.

For the domain portability, K-QARD is designed as a domain-independent architecture and it keeps all domain-dependent elements in external resources. In addition, the framework tries to employ various techniques for reducing the human effort, such as simplifying rules based on linguistic information and machine learning approaches.

Our effort for the robustness is focused the question analysis. Instead of using a technique for deep understanding of the question, the question analysis component of K-QARD tries to extract only essential information for answering using the information extraction technique with linguistic information. Such approach is helpful for not only the robustness but also the domain portability because it generally requires smaller size of hand-crafted rules than a complex semantic grammar.

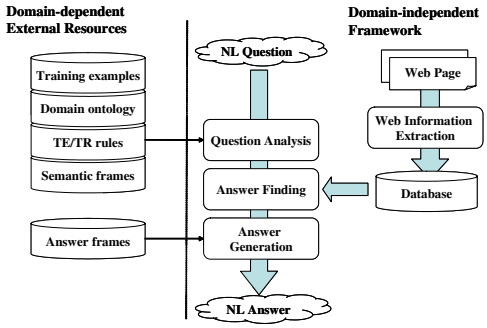


Figure 1: Architecture of K-QARD

K-QARD uses the structural information automatically extracted from Web pages which include domain-specific information for question answering. It has the disavantage that the coverage of QA system is limited, but it can simplify the question answering process with robust performance.

2 Architecture of K-QARD

As shown in Figure 1, K-QARD has four major components: Web information extraction, question analysis, answer finding, and answer generation.

The Web information extraction (IE) component extracts the domain-specific information for question answering from Web pages and stores the information into the relational database. For the domain portability, the Web IE component is based on the automatic wrapper induction approach which can be learned from small size of training examples.

The question analysis component analyzes an

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 29–32,
Sydney, July 2006.

input question, extracts important information using the IE approach, and matches the question with pre-defined semantic frames. The component outputs the best-matched frame whose slots are filled with the information extracted from the question.

In the answer finding component, K-QARD retrieves the answers from the database using the SQL generation script defined in each semantic frame. The SQL script dynamically generates SQL using the values of the frame slots.

The answer generation component provides the answer to the user as a natural language sentence or a table by using the generation rules and the answer frames which consist of canned texts.

3 Question Analysis

The key component for ensuring the robustness and domain portability is the question analysis because it naturally requires many domain-dependent resources and has responsibility to solve the problem caused by various ways of expressing a question. In K-QARD, a question is analyzed using the methods devised by the information extraction approach. This IE-based question analysis method consists of several steps:

1. **Natural language analysis**: Analyzing the syntactic structure of the user's question and also identifiying named-entities and some important words, such as domain-specific predicate or terms.

2. **Question focus recognition**: Finding the intention of the user's question using the question focus classifier. It is learned from the training examples based on decision tree(C4.5)(Quinlan, 1993).

3. **Template Element(TE) recognition**: Finding important concept for filling the slots of the semantic frame, namely template elements, using the rules, NE information, and ontology, etc.

4. **Template Relation(TR) recognition**: Finding the relation between TEs and a question focus based on TR rules, and syntactic information, etc.

Finally, the question analysis component selects the proper frame for the question and fills proper values of each slot of the selected frame.

Compared to other question analysis methods such as the complex semantic grammar(Martin et al., 1996), our approach has several advantages. First, it shows robust performance for the variation of a question because IE-based approach does not require the understanding of the entire sentence. It is sufficient to identify and process only the important concepts. Second, it also enhances the portability of the QA systems. This method is based on the divide-and-conquer strategy and uses only limited context information. By virture of these characteristics, the question analysis can be processed by using a small number of simple rules.

In the following subsections, we will describe each component of our question analyzer in K-QARD.

3.1 Natural language analysis

The natural language analyzer in K-QARD identifies morphemes, tags part-of-speeches to them, and analyzes dependency relations between the morphemes. A stochastic part-of-speech tagger and dependency parser(Chung and Rim, 2004) for the Korean language are trained on a general domain corpus and are used for the analyzer. Then, several domain-specific named entities, such as a TV program name, and general named entities, such as a date, in the question are recognized using our dictionary and pattern-based named entity tagger(Lee et al., 2004). Finally some important words, such as domain-specific predicates, terminologies or interrogatives, are replaced by the proper concept names in the ontology. The manually constructed ontology includes two different types of information: domain-specific and general domain words.

The role of this analyzer is to analyze user's question and transform it to the more generalized representation form. So, the task of the question focus recognition and the TE/TR recognition can be simplified because of the generalized linguistic information without decreasing the performance of the question analyzer.

One of possible defects of using such linguistic information is the loss of the robustness caused by the error of the NLP components. However, our IE-based approach for question analysis uses the very restricted and essential contextual information in each step and can avoid such a risk successfully.

The example of the analysis process of this

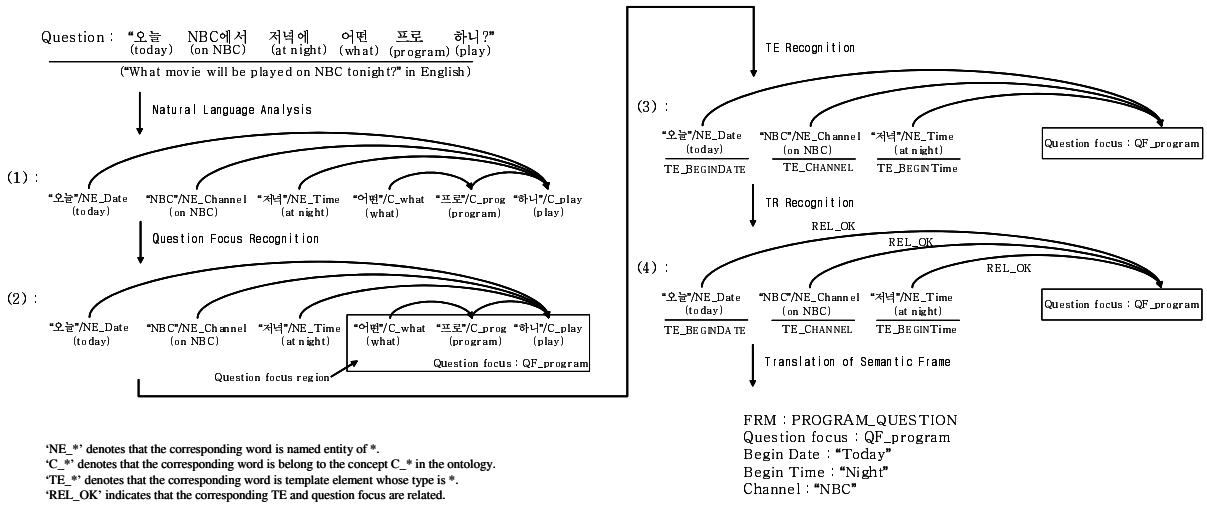


Figure 2: Example of Question Analysis Process in K-QARD

component is shown in Figure 2-(1).

3.2 Question focus recognition

We define a question focus as a type of information that a user wants to know. For example, in the question *"What movies will be shown on TV tonight?"*, the question focus is a program title, or titles. For another example, the question focus is a current rainfall in a question *"San Francisco is raining now, is it raining in Los Angeles too?"*.

To find the question focus, we define *question focus region*, a part of a question that may contain clues for deciding the question focus. The question focus region is identified with a set of simple rules which consider the characteristic of the Korean interrogatives. Generally, the question focus region has a fixed pattern that is typically used in interrogative questions(Akiba et al., 2002). Thus a small number of simple rules is enough to cover the most of question focus region pattern. Figure 2-(2) shows the part recognized as a question focus region in the sample question.

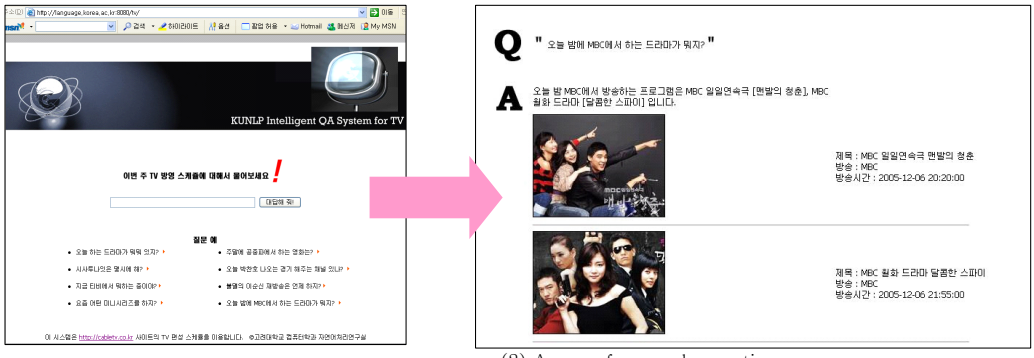
After recognizing the region, the actual focus of the question is determined with features extracted from the question focus region. For the detection, we build the question focus classifier using decision tree (C4.5) and several linguistic or domain-specific features such as the kind of the interrogative and the concept name of the predicate.

Dividing the focus recognition process into two parts helps to increase domain portability. While the second part of deciding the actual question focus is domain-dependent because every domain-application has its own set of question foci, the first part that recognizes the question focus region is domain-independent.

3.3 TE recognition

In the TE identification phase, pre-defined words, phrases, and named entities are identified as slot-filler candidates for appropriate slots, according to TE tagging rules. For instance, *movie* and *NBC* are tagged as Genre and Channel in the sample question *"Tell me the movie on NBC tonight."* (i.e. *movie* will be used to fill Genre slot and *NBC* will be used to fill Channel slot in a semantic frame). The hand-crafted TE tagging rules basically consider the surface form and the concept name (derived from domain ontologies) of a target word. The context surrounding the target word or word dependency information is also considered in some cases. In the example question of Figure 2, the date expression *'오늘(today)'*, time expression *'저녁(night)'* and the channel name *'MBC'* are selected as TE candidates.

In K-QARD, such identification is accomplished by a set of simple rules, which only examines the semantic type of each constituent word in the question, except the words in the question region. It is mainly because of our divide-and-conquer strategy motivated by IE. The result of this component may include some wrong template elements, which do not have any relation to the user's intention or the question focus. However, they are expected to be removed in the next component, the TR recognizer which examines the relation between the recognized TE and the question focus.



(1) Broadcast−domain QA system

(2) Answer for sample question, "What soap opera will be played on MBC tonight?"

Figure 3: Broadcast-domain QA System using K-QARD

3.4 TR recognition

In the TR recognition phase, all entities identified in the TE recognition phase are examined whether they have any relationships with the question focus region of the question. For example, in the question *"Is it raining in Los Angeles like in San Francisco?"*, both *Los Angeles* and *San Francisco* are identified as a TE. However, by the TR recognition, only *Los Angeles* is identified as a related entity with the question focus region.

Selectional restriction and dependency relations between TEs are mainly considered in TR tagging rules. Thus, the TR rules can be quite simplified. For example, many relations between the TEs and the question region can be simply identified by examining whether there is a syntactic dependency between them as shown in Figure 2-(4). Moreover, to make up for the errors in dependency parsing, lexico-semantic patterns are also encoded in the TR tagging rules.

4 Application of K-QARD

To evaluate the K-QARD framework, we built restricted domain question answering systems for the several domains: *weather, broadcast,* and *traffic*. For the adaptation of QA system to each domain, we rewrote the domain ontology consisting of about 150 concepts, about 30 TE/TR rules, and 7-23 semantic frames and answer templates. In addition, we learned the question focus classifier from training examples of about 100 questions for the each domain. All information for the question answering was automatically extracted using the Web IE module of K-QARD, which was also learned from training examples consisting of several annotated Web pages of the target Web site. It took about a half of week for two graduate students who clearly understood the framework to build each QA system. Figure 3 shows an example of QA system applied to the broadcast domain.

5 Conclusion

In this paper, we described the Korean question answering framework, namely K-QARD, for restricted domains. Specifically, this framework is designed to enhance the robustness and domain portability. To achieve this goal, we use the IE-based question analyzer using the generalized information acquired by several NLP components. We also showed the usability of K-QARD by successfully applying the framework to several domains.

References

T. Akiba, K. Itou, A. Fujii, and T Ishikawa. 2002. Towards speech-driven question answering: Experiments using the NTCIR-3 question answering collection. In *Proceedings of the Third NTCIR Workshop*.

H. Chung and H. Rim. 2004. Unlexicalized dependency parser for variable word order languages based on local contextual pattern. *Lecture Note in Computer Science*, (2945):112–123.

J. Lee, Y. Song, S. Kim, H. Chung, and H. Rim. 2004. Title recognition using lexical pattern and entity dictionary. In *Proceedings of the 1st Asia Information Retrieval Symposium (AIRS2004)*, pages 345–348.

P. Martin, F. Crabbe, S. Adams, E. Baatz, and N. Yankelovich. 1996. Speechacts: a spoken language framework. *IEEE Computer*, 7(29):33–40.

J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

An Intermediate Representation for the Interpretation of Temporal Expressions

Paweł Mazur and Robert Dale
Centre for Language Technology
Macquarie University
NSW 2109 Sydney Australia
{mpawel,rdale}@ics.mq.edu.au

Abstract

The interpretation of temporal expressions in text is an important constituent task for many practical natural language processing tasks, including question-answering, information extraction and text summarisation. Although temporal expressions have long been studied in the research literature, it is only more recently, with the impetus provided by exercises like the ACE Program, that attention has been directed to broad-coverage, implemented systems. In this paper, we describe our approach to intermediate semantic representations in the interpretation of temporal expressions.

1 Introduction

In this paper, we are concerned with the interpretation of temporal expressions in text: that is, given an occurrence in a text of an expression like that marked in italics in the following example, we want to determine what point in time is referred to by that expression.

(1) We agreed that we would meet at *3pm on the first Tuesday in November*.

In this particular case, we need to make use of the context of utterance to determine which November is being referred to; this might be derived on the basis of the date stamp of the document containing this sentence. Then we need to compute the full time and date the expression corresponds to. If the utterance in (1) was produced, say, in July 2006, then we might expect the interpretation to be equivalent to the ISO-format expression 2006-11-07T15:00.[1] The derivation of such interpretation was the focus of the TERN evaluations held under the ACE program. Several teams have developed systems which attempt to interpret both simple and much more complex temporal expressions; however, there is very little literature that describes in any detail the approaches taken. This may be due to a perception that such expressions are relatively easy to identify and interpret using simple patterns, but a detailed analysis of the range of temporal expressions that are covered by the TIDES annotation guidelines demonstrates that this is not the case. In fact, the proper treatment of some temporal expressions requires semantic and pragmatic processing that is considerably beyond the state of the art.

Our view is that it is important to keep in mind a clear distinction between, on the one hand, the conceptual model of temporal entities that a particular approach adopts; and, on the other hand, the specific implementation of that model that might be developed for a particular purpose. In this paper, we describe both our underlying framework, and an implementation of that framework. We believe the framework provides a basis for further development, being independent of any particular implementation, and able to underpin many different implementations. By clearly separating the underlying model and its implementation, this also opens the door to clearer comparisons between different approaches.

We begin by summarising existing work in the area in Section 2; then, in Section 3, we describe our underlying model; in Section 4, we describe how this model is implemented in the DANTE

[1] Clearly, other aspects of the document context might suggest a different year is intended; and we might also add the time zone to this value.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 33–36,
Sydney, July 2006.

system.[2]

2 Relation to Existing Work

The most detailed system description in the published literature is that of the Chronos system from ITC-IRST (Negri and Marseglia, 2005). This system uses a large set of hand-crafted rules, and separates the recognition of temporal expressions from their interpretation. The ATEL system developed by the Center for Spoken Language Research (CSLR) at University of Colorado (see (Hacioglu et al., 2005)) uses SVM classifiers to detect temporal expressions. Alias-i's LingPipe also reported results for extraction, but not interpretation, of temporal expressions at TERN 2004.

In contrast to this collection of work, which comes at the problem from a now-traditional information extraction perspective, there is also of course an extensive prior literature on the semantic of temporal expressions. Some more recent work attempts to bridge the gap between these two related enterprises; see, for example, Hobbs and Pan (2004).

3 The Underlying Model

We describe briefly here our underlying conceptual model; a more detailed description is provided in (Dale and Mazur, 2006).

3.1 Processes

We take the ultimate goal of the interpretation of temporal expressions to be that of computing, for each temporal expression in a text, the point in time or duration that is referred to by that expression. We distinguish two stages of processing:

Recognition: the process of identifying a temporal expression in text, and determining its extent.

Interpretation: given a recognised temporal expression, the process of computing the value of the point in time or duration referred to by that expression.

In practice, the processes involved in determining the extent of a temporal expression are likely to make use of lexical and phrasal knowledge that mean that some of the semantics of the expression can already be computed. For example, in order to identify that an expression refers to a day of the week, we will in many circumstances need to recognize whether one of the specific expressions {*Monday*, *Tuesday*, ... *Sunday*} has been used; but once we have recognised that a specific form has been used, we have effectively computed the semantics of that part of the expression.

To maintain a strong separation between recognition and interpretation, one could simply recompute this partial information in the interpretation phase; this would, of course, involve redundancy. However, we take the view that the computation of partial semantics in the first step should not be seen as violating the strong separation; rather, we distinguish the two steps of the process in terms of the extent to which they make use of contextual information in computing values. Then, recognition is that phase which makes use only of expression-internal information and preposition which precedes the expression in question; and interpretation is that phase which makes use of arbitrarily more complex knowledge sources and wider document context. In this way, we motivate an intermediate form of representation that represents a 'context-free' semantics of the expression.

The role of the recognition process is then to compute as much of the semantic content of a temporal expression as can be determined on the basis of the expression itself, producing an intermediate partial representation of the semantics. The role of the interpretation process is to 'fill in' any gaps in this representation by making use of information derived from the context.

3.2 Data Types

We view the temporal world as consisting of two basic types of entities, these being **points in time** and **durations**; each of these has an internal hierarchical structure. It is convenient to represent these as feature structures like the following:[3]

(2)
$$\begin{bmatrix} \textit{point} & & \\ \text{DATE} & \begin{bmatrix} \text{DAY} & \textit{11} \\ \text{MONTH} & \textit{6} \\ \text{YEAR} & \textit{2005} \end{bmatrix} \\ \text{TIME} & \begin{bmatrix} \text{HOUR} & \textit{3} \\ \text{MINUTE} & \textit{00} \\ \text{AMPM} & \textit{pm} \end{bmatrix} \end{bmatrix}$$

[2] DANTE stands for Detection and Normalisation of Temporal Expressions.

[3] For reasons of limitations of space, we will ignore durations in the present discussion; their representation is similar in spirit to the examples provided here.

Our choice of attribute–value matrices is not accidental; in particular, some of the operations we want to carry out on the interpretations of both partial and complete temporal expressions can be conveniently expressed via unification, and this representation is a very natural one for such operations.

This same representation can be used to indicate the interpretation of a temporal expression at various stages of processing, as outlined below. In particular, note that temporal expressions differ in their **explicitness**, i.e. the extent to which the interpretation of the expression is explicitly encoded in the temporal expression; they also differ in their **granularity**, i.e. the smallest temporal unit used in defining that point in time or duration. So, for example, in a temporal reference like *November 11th*, interpretation requires us to make explicit some information that is not present (that is, the year); but it does not require us to provide a time, since this is not required for the granularity of the expression.

In our attribute–value matrix representation, we use a special NULL value to indicate granularities that are not required in providing a full interpretation; information that is not explicitly provided, on the other hand, is simply absent from the representation, but may be added to the structure during later stages of interpretation. So, in the case of an expression like *November 11th*, the recognition process may construct a partial interpretation of the following form:

(3)
$$\begin{bmatrix} \textit{point} & \\ \text{DATE} & \begin{bmatrix} \text{DAY} & \textit{11} \\ \text{MONTH} & \textit{6} \end{bmatrix} \\ \text{TIME} & \text{NULL} \end{bmatrix}$$

The interpretation process may then monotonically augment this structure with information from the context that allows the interpretation to be made fully explicit:

(4)
$$\begin{bmatrix} \textit{point} & \\ \text{DATE} & \begin{bmatrix} \text{DAY} & \textit{11} \\ \text{MONTH} & \textit{6} \\ \text{YEAR} & \textit{2006} \end{bmatrix} \\ \text{TIME} & \text{NULL} \end{bmatrix}$$

The representation thus very easily accommodates relative underspecification, and the potential for further specification by means of unification, although our implementation also makes use of other operations applied to these structures.

4 Implementation

4.1 Data Structures

We could implement the model above directly in terms of recursive attribute–value structures; however, for our present purposes, it turns out to be simpler to implement these structures using a string-based notation that is deliberately consistent with the representations for values used in the TIMEX2 standard (Ferro et al., 2005). In that notation, a time and date value is expressed using the ISO standard; uppercase Xs are used to indicate parts of the expression for which interpretation is not available, and elements that should not receive a value are left null (in the same sense as our NULL value above). So, for example, in a context where we have no way of ascertaining the century being referred to, the TIMEX2 representation of the value of the underlined temporal expression in the sentence *We all had a great time in the ’60s* is simply `VAL="XX6"`.

We augment this representation in a number of ways to allow us to represent intermediate values generated during the recognition process; these extensions to the representation then serve as means of indicating to the interpretation process what operations need to be carried out.

4.1.1 Representing Partial Specification

We use lowercase xs to indicate values that the interpretation process is required to seek a value for; and by analogy, we use a lowercase t rather than an uppercase T as the date–time delimiter in the structure to indicate when the recogniser is not able to determine whether the time is am or pm. This is demonstrated in the following examples; `T-VAL` is the attribute we use for intermediate TIMEX values produced by the recognition process.

(5) a. We’ll see you in *November*.
b. `T-VAL="xxxx-11"`

(6) a. I expect to see you at *half past eight*.
b. `T-VAL="xxxx-xx-xxt08:59"`

(7) a. I saw him back in *’69*.
b. `T-VAL="xx69"`

(8) a. I saw him back in the *’60s*.
b. `TVAL="xx6"`

4.1.2 Representing Relative Specification

To handle the partial interpretation of relative date and time expressions at the recognition stage, we

use two extensions to the notation. The first provides for simple arithmetic over interpretations, when combined with a reference date determined from the context:

(9) a. We'll see you *tomorrow*.
b. `T-VAL="+0000-00-01"`

(10) a. We saw him *last year*.
b. `T-VAL="-0001"`

The second provides for expressions where a more complex computation is required in order to determine the specific date or time in question:

(11) a. We'll see him *next Thursday*.
b. `T-VAL=">D4"`

(12) a. We saw him *last November*.
b. `T-VAL="<M11"`

4.2 Processes

For the recognition process, we use a large collection of rules written in the JAPE pattern-matching language provided within GATE (see (Cunningham et al., 2002)). These return intermediate values of the forms described in the previous section. Obviously other approaches to recognizing temporal expressions and producing their intermediate values could be used; in DANTE, there is also a subsequent check carried out by a dependency parser to ensure that we have captured the full extent of the temporal expression.

DANTE's interpretation process then does the following. First it determines if the candidate temporal expression identified by the recogniser is indeed a temporal expression; this is to deal with cases where a particular word or phrase annotated by the recognizer (such as *time*) can have both temporal or non-temporal interpretations. Then, for each candidate that really is a temporal expression, it computes the interpretation of that temporal expression.

This second step involves different operations depending on the type of the intermediate value:

- Underspecified values like `XXXX-11` are combined with the reference date derived from the document context, with temporal directionality (i.e., is this date in the future or in the past?) being determined using tense information from the host clause.
- Relative values like `+0001` are combined with the reference date in the obvious manner.
- Relative values like `>D4` and `<M11` make use of special purpose routines that know about arithmetic for days and months, so that the correct behaviour is observed.

5 Conclusions

We have sketched an underlying conceptual model for temporal expression interpretation, and presented an intermediate semantic representation that is consistent with the TIMEX2 standard. We are making available a corpus of examples tagged with these intermediate representations; this corpus is derived from the nearly 250 examples in the TIMEX2 specification, thus demonstrating the wide coverage of the representation. Our hope is that this will encourage collaborative development of tools based on this framework, and further development of the conceptual framework itself.

6 Acknowledgements

We acknowledge the support of DSTO, the Australian Defence Science and Technology Organisation, in carrying out the work described here.

References

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the ACL*.

R. Dale and P. Mazur. 2006. Local semantics in the interpretation of temporal expressions. In *Proceedings of the Coling/ACL2006 Workshop on Annotating and Reasoning about Time and Events*.

L. Ferro, L. Gerber, I. Mani, B. Sundheim, and G. Wilson. 2005. TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE, September.

K. Hacioglu, Y. Chen, and B. Douglas. 2005. Automatic Time Expression Labeling for English and Chinese Text. In Alexander F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 6th International Conference, CICLing'05*, LNCS, pages 548–559. Springer.

Jerry R. Hobbs and Feng Pan. 2004. An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing*, 3(1), March.

M. Negri and L. Marseglia. 2005. Recognition and Normalization of Time Expressions: ITC-irst at TERN 2004. Technical Report WP3.7, Information Society Technologies, February.

Chinese Named Entity and Relation Identification System

Tianfang Yao
Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai, 200030, China
yao-tf@cs.sjtu.edu.cn

Hans Uszkoreit
Department of Computational Linguistics and Phonetics
Saarland University
Saarbrücken, 66041, Germany
uszkoreit@coli.uni-sb.de

Abstract

In this interactive presentation, a Chinese named entity and relation identification system is demonstrated. The domain-specific system has a three-stage pipeline architecture which includes word segmentation and part-of-speech (POS) tagging, named entity recognition, and named entity relation identitfication. The experimental results have shown that the average F-measure for word segmentation and POS tagging after correcting errors achieves 92.86 and 90.01 separately. Moreover, the overall average F-measure for 6 kinds of name entities and 14 kinds of named entity relations is 83.08% and 70.46% respectively.

1 Introduction

The investigation for Chinese information extraction is one of the topics of the project COLLATE (DFKI, 2002) dedicated to building up the German Competence Center for Language Technology. The presented work aims at investigating automatic identification of Chinese named entities (NEs) and their relations in a specific domain.

Information Extraction (IE) is an innovative language technology for accurately acquiring crucial information from documents. NE recognition is a fundamental IE task, that detects some named constituents in sentences, for instance names of persons, places, organizations, dates, times, and so on. Based on NE recognition, the identification of Named Entity Relation (NER) can indicate the types of semantic relationships between identified NEs. e.g., relationships between person and employed organization; person and residing place; person and birthday; organization and seat, etc. The identified results for NEs and NERs can be provided as a resource for other application systems such as question-answering system. Therefore, these two IE tasks are selected as our investigation emphases.

Chinese has a very different structure from western languages. For example, it has a large character set involving more than 48,000 characters; there is no space between words in written texts; and Chinese words have fewer inflections, etc. In the past twenty years there have been significant achievements in IE concerning western languages such as English. Comparing with that, the research on the relevant properties of Chinese for IE, especially for NER, is still insufficient.

Our research focuses on domain-specific IE. We picked the sports domain, particularly, texts on soccer matches because the number and types of entities, relations and linguistic structures are representative for many applications.

Based on the motivations above mentioned, our goals for the design and implementation of the prototype system called CHINERIS (Chinese Named Entity and Relation Identification System) are:

- Establishing an IE computational model for Chinese web texts using hybrid technologies, which should to a great extent meet the requirements of IE for Chinese web texts;
- Implementing a prototype system based on this IE computational model, which extracts information from Chinese web texts as accurately and quickly as possible;
- Evaluating the performance of this sys tem in a specific domain.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 37–40,
Sydney, July 2006.

2 System Design

In the model, the IE processing is divided into three stages: (i) word segmentation and part-of-speech (POS) tagging; (ii) NE recognition; (iii) NER identification. Figure 1 demonstrates a Chinese IE computational model comprised of these three stages. Each component in the system corresponds to a stage.

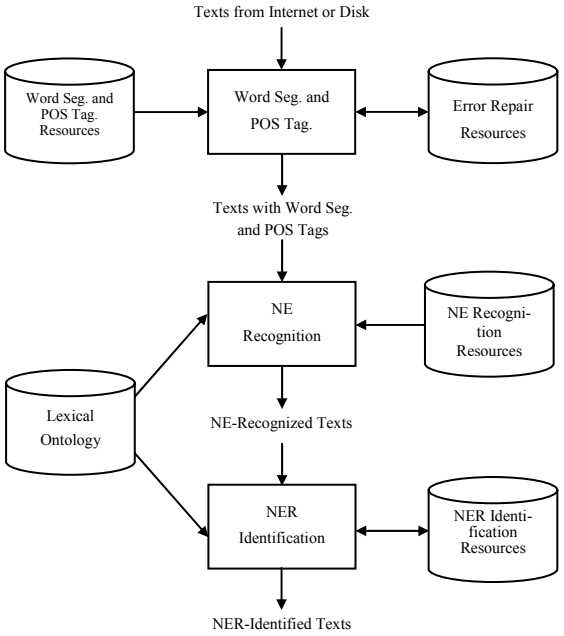


Figure 1. A three-stage Chinese IE computational model.

In general, the accuracy of the first stage has considerable influence on the performance of the consequent two stages. It has been demonstrated by our experiments (Yao et al., 2002). In order to reduce unfavorable influence, we utilize a trainable approach (Brill, 1995) to automatically generate effective rules, by which the first component can repair different errors caused by word segmentation and POS tagging.

At the second stage, there are two kinds of NE constructions to be processed (Yao et al., 2003). One is the NEs which involve trigger words; the other those without trigger words. For the former NEs, a shallow parsing mechanism, i.e., finite-state cascades (FSC) (Abney, 1996) which are automatically constructed by sets of NE recognition rules, is adopted for reliably identifying different categories of NEs. For the latter NEs, however, some special strategies, such as the valence constraints of domain verbs, the constituent analysis of NE candidates, the global context clues and the analysis for preposition objects etc., are designed for identifying them.

After the recognition for NEs, NER identification is performed in the last stage. Because of the diversity and complexity of NERs, at the same time, considering portability requirement in the identification, we suggest a novel supervised machine learning approach called positive and negative case-based learning (PNCBL) used in this stage (Yao and Uszkoreit, 2005).

The learning in this approach is a variant of memory-based learning (Daelemans et al., 2000). The goal of that is to capture valuable information from NER and non-NER patterns, which is implicated in different features. Because not all features we predefine are necessary for each NER or non-NER, we should select them by a reasonable measure mode. According to the selection criterion we propose - self-similarity, which is a quantitative measure for the concentrative degree of the same kind of NERs or non-NERs in the corresponding pattern library, the effective feature sets - General-Character Feature (GCF) sets for NERs and Individual-Character Feature (ICF) sets for non-NERs are built. Moreover, the GCF and ICF feature weighting serve as a proportion determination of feature's degree of importance for identifying NERs against non-NERs. Subsequently, identification thresholds can also be determined.

Therefore, this approach pursues the improvement of the identification performance for NERs by simultaneously learning two opposite cases, automatically selecting effective multi-level linguistic features from a predefined feature set for each NER and non-NER, and optimally making an identification tradeoff. Further, two other strategies, resolving relationship conflicts and inferring missing relationships, are also integrated in this stage.

Considering the actual requirements for domain knowledge, we defined a hierarchical taxonomy and constructed conceptual relationships among Object, Movement and Property concept categories under the taxonomy in a lexical sports ontology (Yao, 2005). Thus, this ontology can be used for the recognition of NEs with special constructions - without trigger words, the determination of NE boundaries, and the provision of feature values as well as the computation of the semantic distance for two concepts during the identification of NERs.

3 System Implementation

During the implementation, object-oriented design and programming methods are thoroughly

used in the system development. In order to avoid repeated development, we integrate other application system and resource, e.g., Modern Chinese Word Segmentation and POS Tagging System (Liu, 2000) and HowNet (Dong and Dong, 2000) into the system. Additionally, we utilize Protégé-2000 (version 1.9) (Stanford Medical Informatics, 2003) as a development environment for the implementation of lexical sports ontology.

The prototype system CHINERIS has been implemented in Java. The system can automatically identify 6 types of NEs[1] and 14 types of NERs[2] in the sports domain. Furthermore, its run-time efficiency is acceptable and the system user interfaces are friendly.

4 Testing and Evaluation

We have finished three experiments for testing three components. Table 1 shows the experimental results for the performance of these components.

Stage	Task	(Total) Ave. Rec.	(Total) Ave. Pre.	(Total) Ave. F-M
1st	Word Seg.	95.08	90.74	92.86
	POS Tag.	92.39	87.75	90.01
2nd	NE Ident.	83.38	82.79	83.08
3rd	NER Ident.	78.50	63.92	70.46

Table 1. Performance for the System CHINERIS.

In the first experiment, the training set consists of 94 texts including 3473 sentences collected from the soccer matches of the Jie Fang Daily (http://www.jfdaily.com/) in 2001. During manual error-correction, we adopted a double-person annotation method. After training, we obtain error repair rules. They can repair at least one error in the training corpus. The rules in the rule library are ranked according to the errors they correct. The testing set is a separate set that contains 20 texts including 658 sentences. The texts in the testing set have been randomly chosen from the Jie Fang Daily from May 2002. In the testing, the usage of error repair rules with context constraints has priority over those without context constraints, and the usage of error repair rules for word segmentation has priority over those for POS tagging. Through experimental observation, this processing sequence can ensure that the rules repair many more errors. On the other hand, it can prevent new errors occurring during the repair of existing errors. The results indicate that after the correction, the average F-measure of word segmentation has increased from 87.75 % to 92.86%; while that of POS tagging has even increased from 77.47% to 90.01%. That is to say, the performance of both processes has been distinctly enhanced.

In the second experiment, we utilize the same testing set for the error repair component to check the named entity identification which includes regular and special entity constructions. The rule sets provided for TN, CT, and PI recognition have 35, 50, and 20 rules respectively. In lexical sports ontology, there are more than 350 domain verbs used for the identification of TN with special constructions. Among six NEs, the average F-measure of DT, PI, and CT exceeds 85%. Therefore, it specifies that the identification performance of named entities after adding the special recognition strategies in this component has reached a good level.

In the third experiment, both pattern libraries are established in terms of the annotated texts and lexical sports ontology during learning. They have 142 (534 NERs) and 98 (572 non-NERs) sentence groups respectively. To test the performance of our approach, we randomly choose 32 sentence groups from the Jie Fang Daily in 2002 (these sentence groups are out of either NER or non-NER pattern library), which embody 117 different NER candidates. Table 1 shows the total average recall, precision, and F-measure for 14 different NERs by positive and negative case-based learning and identification. Among 14 types of NERs, the highest total average F-measure is 95.65 from the relation LOC_CPC and the lowest total average F-measure is 34.09 from TM_CPC. The total average F-measure is 70.46. In addition, we also compared the performance between the total average recall, precision, and F-measure for all NERs only by positive and by positive and negative case-based learning and identification separately. It shows the total average F-measure is enhanced from 63.61% to 70.46% as a whole,

[1] Personal Name (PN); Date or Time (DT); Location Name (LN); Team Name (TN); Competition Title (CT); Personal Identity (PI).

[2] Person ↔ Team (PS_TM); Person ↔ Competition (PS_CP); Person ↔ City / Province / Country (PS_CPC); Person ↔ Identification (PS_ID); Home Team ↔ Visiting Team (HT_VT); Winning Team ↔ Losing Team (WT_LT); Draw Team ↔ Draw Team (DT_DT); Team ↔ Competition (TM_CP); Team ↔ City / Province / Country (TM_CPC); Identification ↔ Team (ID_TM); Competition ↔ Date (CP_DA); Competition ↔ Time (CP_TI); Competition ↔ Location (CP_LOC); Location ↔ City / Province / Country (LOC_ CPC).

due to the adoption of both positive and negative cases.

From the result, we also realize that the selection of relation features is critical. First, they should be selected from multiple linguistic levels, e.g., morphology, syntax and semantics. Second, they should also embody the crucial information of Chinese language processing, such as word order, the context of words, and particles etc. Moreover, the proposed self-similarity is a reasonable measure for selecting GCF and ICF for NERs and non-NERs identification respectively.

5 Conclusion

This three-stage IE prototype system CHINERIS is appropriate and effective for Chinese named entity and relation identification in sports domain.

In the first component, it is a beneficial exploration to develop an error repairer which simultaneously enhances the performance of Chinese word segmentation and POS tagging.

In the second component, we theoretically extend the original definition of Finite State Automata (FSA), that is, we use complex constraint symbols rather than atomic constraint symbols. With this extension, we improve the practicability for the FSC mechanism. At the same time, the new issue for automatically constructing FSC also increases the flexibility of its maintenance. In order to improve the NE identification performance, some special strategies for the identification of NEs without trigger words are added in this stage, which cannot be recognized by FSC.

In the third component, automatically selecting effectual multi-level linguistic features for each NER and non-NER and learning two opposite types of cases simultaneously are two innovative points in the PNCBL approach.

The lexical sports ontology plays an important role in the identification of NEs and NERs, such as determination of the boundary of NEs, identification for NE with special constructons and calculation of similarity for the features (e.g. semantic distance).

The experimental results for the three components in the prototype system show that the system CHINERIS is successful for the sample application.

Acknowledgement

This work is a part of the COLLATE project under contract no. 01INA01B, which is supported by the German Ministry for Education and Research.

References

S. Abney. 1996. *Partial Parsing via Finite-State Cascades*. In Proceedings of the ESSLLI '96 Robust Parsing Workshop, pages 8-15. Prague, Czech Republic.

E. Brill. 1995. *Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging*. Computational Linguistics, 21(4): 543-565.

W. Daelemans, A. Bosch, J. Zavrel, K. Van der Sloot, and A. Vanden Bosch. 2000. *TiMBL: Tilburg Memory Based Learner*, Version 3.0, Reference Guide. Technical Report ILK-00-01, ILK, Tilburg University. Tilburg, The Netherlands. http://ilk.kub.nl/~ilk/papers/ilk0001.ps.gz.

DFKI. 2002. *COLLATE: Computational Linguistics and Language Technology for Real Life Applications*. DFKI, Saarbrücken, Germany. http://collate.dfki. de/.

Z. Dong and Q. Dong. 2000. *HowNet*. http://www.keenage.com/zhiwang/e_zhiwang.html.

K. Liu. 2000. *Automatic Segmentation and Tagging for Chinese Text*. The Commercial Press. Beijing, China.

Stanford Medical Informatics. 2003. *The Protégé Ontology Editor and Knowledge Acquisition System*. The School of Medicine, Stanford University. Stanford, USA. http://protege.stanford.edu/.

T. Yao, W. Ding, and G. Erbach. 2002. *Correcting Word Segmentation and Part-of-Speech Tagging Errors for Chinese Named Entity Recognition*. In G. Hommel and H. Sheng, editors, The Internet Challenge: Technology and Applications, pages 29-36. Kluwer Academic Publishers. The Netherlands.

T. Yao, W. Ding and G. Erbach. 2003. *CHINERS: A Chinese Named Entity Recognition System for the Sports Domain*. In: Proc. of the Second SIGHAN Workshop on Chinese Language Processing (ACL 2003 Workshop), pages 55-62. Sapporo, Japan.

T. Yao and H. Uszkoreit. 2005. *A Novel Machine Learning Approach for the Identification of Named Entity Relations*. In: Proc. of the Workshop on Feature Engineering for Machine Learning in Natural Language Processing (ACL 2005 Workshop), pages 1-8. Michigan, USA.

T. Yao. 2005. *A Lexical Ontology for Chinese Information Extraction*. In M. Sun and Q. Chen, editors, Proc. of the 8th National Joint Symposium on Computational Linguistics (JSCL-2005), pages 241-246. Nanjing, China.

Computational Analysis of Move Structures in Academic Abstracts

Jien-Chen Wu[1] Yu-Chia Chang[1] Hsien-Chin Liou[2] Jason S. Chang[1]
CS[1] and FLL[2], National Tsing Hua Univ.
{d928322,d948353}@oz.nthu.edu.tw, hcliu@mx.nthu.edu.tw,
jason.jschang@gmail.com

Abstract

This paper introduces a method for computational analysis of move structures in abstracts of research articles. In our approach, sentences in a given abstract are analyzed and labeled with a specific move in light of various rhetorical functions. The method involves automatically gathering a large number of abstracts from the Web and building a language model of abstract moves. We also present a prototype concordancer, CARE, which exploits the move-tagged abstracts for digital learning. This system provides a promising approach to Web-based computer-assisted academic writing.

1 Introduction

In recent years, with the rapid development of globalization, English for Academic Purposes has drawn researchers' attention and become the mainstream of English for Specific Purposes, particularly in the field of English of Academic Writing (EAW). EAW deals mainly with genres, including research articles (RAs), reviews, experimental reports, and other types of academic writing. RAs play the most important role of offering researchers the access to actively participating in the academic and discourse community and sharing academic research information with one another.

Abstracts are constantly regarded as the first part of RAs and few scholarly RAs go without an abstract. “A well-prepared abstract enables readers to identify the basic content of a document quickly and accurately.” (American National Standards Institute, 1979) Therefore, RAs' abstracts are equally important to writers and readers.

Recent research on abstract requires manually analysis, which is time-consuming and labor-intensive. Moreover, with the rapid development of science and technology, learners are increasingly engaged in self-paced learning in a digital environment. Our study, therefore, attempts to investigate ways of automatically analyzing the move structure of English RAs’ abstracts and develops an online learning system, CARE (Concordancer for Academic wRiting in English). It is expected that the automatic analytical tool for move structures will facilitate non-native speakers (NNS) or novice writers to be aware of appropriate move structures and internalize relevant knowledge to improve their writing.

2 Macrostructure of Information in RAs

Swales (1990) presented a simple and succinct picture of the organizational pattern for a RA—the IMRD structure (Introduction, Methods, Results, and Discussion). Additionally Swales (1981, 1990) introduced the theory of genre analysis of a RA and a four-move scheme, which was later refined as the "Create a Research Space" (CARS) model for analyzing a RA’s introduction section.

Even though Swales seemed to have overlooked the abstract section, in which he did not propose any move analysis, he himself plainly realized “abstracts continue to remain a neglected field among discourse analysts” (Swales, 1990, p. 181). Salager-Meyer (1992) also stated, “Abstracts play such a pivotal role in any professional reading” (p. 94). Seemingly researchers have perceived this view, so research has been expanded to concentrate on the abstract in recent years.

Anthony (2003) further pointed out, “research has shown that the study of rhetorical organization or structure of texts is particularly useful in the technical reading and writing classroom” (p. 185). Therefore, he utilized computational means to create a system, *Mover*, which could offer move analysis to assist abstract writing and reading.

3 CARE

Our system focuses on automatically computational analysis of move structures (i.e.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 41–44,
Sydney, July 2006.

Background, **P**urpose, **M**ethod, **R**esult, and **C**onclusion) in RA abstracts. In particular, we investigate the feasibility of using a few manually labeled data as seeds to train a Markov model and to automatically acquire move-collocation relationships based on a large number of unlabeled data. These relationships are then used to analyze the rhetorical structure of abstracts. It is important that only a small number of manually labeled data are required while much of move tagging knowledge is learned from unlabeled data. We attempt to identify which rhetorical move is correspondent to a sentence in a given abstract by using features (e.g. collocations in the sentence). Our learning process is shown as follows:

(1)Automatically collect abstracts from the Web for training

(2)Manually label each sentence in a small set of given abstracts

(3)Automatically extract collocations from all abstracts

(4)Manually label one move for each distinct collocation

(5)Automatically expand collocations indicative of each move

(6)Develop a hidden Markov model for move tagging

Figure 1: Processes used to learn collocation classifiers

3.1 Collecting Training Data

In the first four processes, we collected data through a search engine to build the abstract corpus *A*. Three specialists in computer science tagged a small set of the qualified abstracts based on our coding scheme of moves. Meanwhile, we extracted the collocations (Jian et al., 2004) from the abstract corpus, and labeled these extracted collocations with the same coding scheme.

3.2 Automatically Expanding Collocations for Moves

To balance the distribution in the move-tagged collocation (*MTC*), we expand the collocation for certain moves in this stage. We use the one-move-per-collocation constraint to bootstrap, which mainly hinges on the feature redundancy of the given data, a situation where there is often evidence to indicate that a given should be annotated with a certain move. That is, given one collocation c_i is tagged with move m_i, all sentences *S* containing collocation c_i will be tagged with m_i as well; meanwhile, the other collocations in *S* are thus all tagged with m_i. For example:

Step 1. The collocation "paper address" extracted from corpus *A* is labeled with the "P" move. Then we use it to label other untagged sentences *US* (e.g. Examples (1) through (2)) containing "paper address" as "P" in *A*. As a result, these *US* become tagged sentences *TS* with "P" move.

(1)This **paper addresses** the state explosion problem in automata based ltl model checking. //P//

(2)This **paper addresses** the problem of fitting mixture densities to multivariate binned and truncated data. //P//

Step 2. We then look for other features (e.g. the collocation, "address problem") that occur in *TS* of *A* to discover new evidences of a "P" move (e.g. Examples (3) through (4)).

(3)This paper **addresses** the state explosion **problem** in automata based ltl model checking.

(4)This paper **addresses** the **problem** of fitting mixture densities to multivariate binned and truncated data.

Step 3. Subsequently, the feature "address problem" can be further exploited to tag sentences which realize the "P" move but do not contain the collocation "paper address", thus gradually expanding the scope of the annotations to *A*. For example, in the second iteration, Example (5) and (6) can be automatically tagged as indicating the "P" move.

(5)In this paper we **address** the **problem** of query answering using views for non-recursive data log queries embedded in a Description Logics knowledge base. //P//

(6)We **address** the **problem** of learning robust plans for robot navigation by observing particular robot behaviors. //P//

From these examples ((5) and (6)), we can extend to another feature "we address", which can be tagged as "P" move as well. The bootstrapping processes can be repeated until no new feature with high enough frequency is found (a sample of collocation expanded list is shown in Table1).

Type	Collocation	Move	Count of Collocation with m_i	Total of Collocation Occurrences
NV	we present	P	3,441	3,668
NV	we show	R	1,985	2,069
NV	we propose	P	1,722	1,787
NV	we describe	P	1,505	1,583
…	…	…	…	…

Table 1: The sample of the expanded collocation list

3.3 Building a HMM for Move Tagging

The move sequence probability $P(t_i{+}1 \mid t_i)$ is given as the following description:

We are given a corpus of unlabeled abstracts $A = \{A_1,\ldots, A_N\}$. We are also given a small labeled subset $S = \{L_1,\ldots, L_k\}$ of A, where each abstract L_i consists of a sequence of sentence and move $\{t_1, t_2,\ldots, t_k\}$. The moves t_i take out of a value from a set of possible move $M = \{m_1,m_2,\ldots,m_n\}$.

$$\text{Then } P(t_{i+1} \mid t_i) = \left(\frac{N(t_i \mid t_{i+1})}{N(t_i)} \right)$$

According to the bi-gram move sequence score (shown in Table 2), we can see move sequences follow a certain schematic pattern. For instance, the "B" move is usually directly followed by the "P" move or "B" move, but not by the "M" move. Also rarely will a "P" move occur before a "B" move. Furthermore, an abstract seldom have a move sequence wherein "P" move directly followed by the "R" move, which tends to be a bad move structure. In sum, the move progression generally follows the sequence of "B-P-M-R-C".

Move t_i	Move t_{i+1}	- log P (t_{i+1}|t_i)
$	B	0.7802
$	P	0.6131
B	B	0.9029
B	M	3.6109
B	P	0.5664
C	$	0.0000
M	$	4.4998
M	C	1.9349
M	M	0.7386
M	R	1.0033
P	M	0.4055
P	P	1.1431
P	R	4.2341
R	$	0.9410
R	C	0.8232
R	R	1.7677

Table 2: The score of bi-gram move sequence (Note that "$" denotes the beginning or the ending of a given abstract.)

Finally, we synchronize move sequence and one-move-per-collocation probabilities to train a language model to automatically learn the relationship between those extracted linguistic features based on a large number of unlabeled data. Meanwhile, we set some parameters of the proposed model, such as, the threshold of the number of collocation occurring in a given abstract, the weight of move sequence and collocation and smoothing. Based on these parameters, we implement the Hidden Markov Model (HMM). The algorithm is described as the following:

$$p(s_1,....,s_n) = p(t_1)p(s_1 \mid t_1)\Pi p(t_i \mid t_{i-1})p(s_i \mid t_i)$$

The moves t_i take out of a value from a set of possible moves $M=\{m_1, m_2, \ldots, m_k\}$ (The following parameters θ_1 and θ_2 will be determined based on some heuristics).

$$p(S_i \mid t_i = m_i)$$
$$= \theta_1 \text{ if } S_i \text{ contains a collocation in } MTC_j \quad i = j$$
$$= \theta_2 \text{ if } S_i \text{ contains a collocation in } MTC_j \text{ but } i \neq j$$
$$= \frac{1}{k} \text{ if } S_i \text{ does not contain a collocation } MTC_j$$

The optimal move sequence t* is

$$(t_1*, t_2*, ..., t_n*) = \arg\max_{t_1,t_2,....,t_n} p(s_1, ..., s_n \mid t_i, ..., t_n)$$

In summary, at the beginning of training time, we use a few human move-tagged sentences as seed data. Then, collocation-to-move and move-to-move probabilities are employed to build the HMM. This probabilistic model derived at the training stage will be applied at run time.

4 Evaluation

In terms of the training data, we retrieved abstracts from the search engine, *Citeseer*; a corpus of 20,306 abstracts (95,960 sentences) was generated. Also 106 abstracts composed of 709 sentences were manually move-tagged by four informants. Meanwhile, we extracted 72,708 collocation types and manually tagged 317 collocations with moves.

At run time, 115 abstracts containing 684 sentences were prepared to be the training data. We then used our proposed HMM to perform some experimentation with the different values of parameters: the frequency of collocation types, the number of sentences with collocation in each abstract, move sequence score and collocation score.

4.1 Performance of CARE

We investigated how well the HMM model performed the task of automatic move tagging under different values of parameters. The parameters involved included the weight of transitional probability function, the number of sentences in an abstract, the minimal number of instance for the applicable collocations. Figure 2 indicates the best precision of 80.54% when 627 sentences were qualified with the set of various

parameters, including 0.7 as the weight of transitional probability function and a frequency threshold of 18 for a collocation to be applicable, and the minimally two sentences containing an applicable collocation. Although it is important to have many collocations, it is crucial that we set an appropriate frequency threshold of collocation so as not to include unreliable collocation and lower the precision rate.

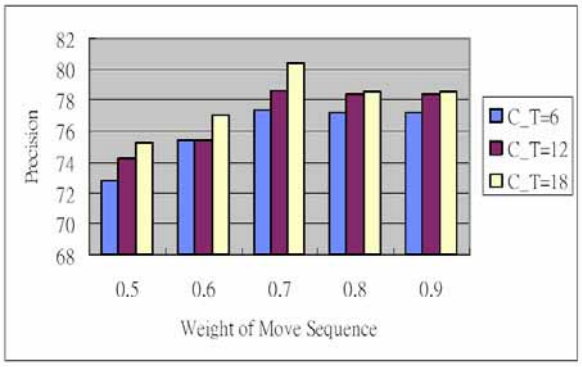


Figure2: The results of tagging performance with different setting of weight and threshold for applicable collocations (Note that C_T denotes the frequency threshold of collocation)

5 System Interface

The goal of the CARE System is to allow a learner to look for instances of sentences labeled with moves. For this purpose, the system is developed with three text boxes for learners to enter queries in English (as shown in Figure3.):

- Single word query (i.e. directly input one word to query)
- Multi-word query (i.e. enter *the result show* to find citations that contain the three words, "*the*", "*paper*" and "*show*" and all the derivatives)
- Corpus selection (i.e. learners can focus on a corpus in a specific domain)

Once a query is submitted, CARE displays the results in returned Web pages. Each result consists of a sentence with its move annotation. The words matching the query are highlighted.

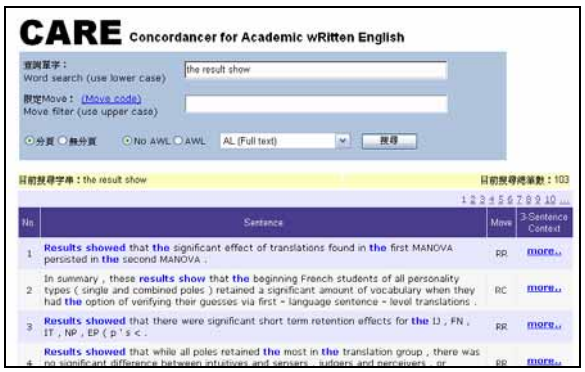


Figure 3: The sample of searching result with the phrase "the result show"

6 Conclusion

In this paper, we have presented a method for computational analysis of move structures in RAs' abstracts and addressed its pedagogical applications. The method involves learning the inter-move relationships, and some labeling rules we proposed. We used a large number of abstracts automatically acquired from the Web for training, and exploited the HMM to tag sentences with the move of a given abstract. Evaluation shows that the proposed method outperforms previous work with higher precision. Using the processed result, we built a prototype concordance, CARE, enriched with words, phrases and moves. It is expected that NNS can benefit from such a system in learning how to write an abstract for a research article.

References

Anthony, L. and Lashkia, G. V. 2003. Mover: A machine learning tool to assist in the reading and writing of technical papers. *IEEE Trans. Prof. Communication*, 46:185-193.

American National Standards Institute. 1979. *American national standard for writing abstracts*. ANSI Z39, 14-1979. New York: Author.

Jian, J. Y., Chang, Y. C., and Chang, J. S. 2004. TANGO: Bilingual Collocational Concordancer, Post & demo in ACL 2004, Barcelona.

Salager-Meyer, F. S. 1992. A text-type and move analysis study of verb tense and modality distribution in medical English abstracts. *English for Specific Purposes*, 11:93-113.

Swales, J.M. 1981. *Aspects of article introductions. Birmingham*, UK: The University of Aston, Language Studies Unit.

Swales, J.M. 1990. *Genre analysis: English in Academic and Research Settings*. Cambridge University Press.

LexNet: A Graphical Environment for Graph-Based NLP

Dragomir R. Radev[1,2], **Güneş Erkan**[1], **Anthony Fader**[3],
Patrick Jordan[1], Siwei Shen[1], and James P. Sweeney[2]
Department of Electrical Engineering and Computer Science
School of Information
Department of Mathematics
University of Michigan
Ann Arbor, MI 48109
{radev, gerkan, afader, prjordan, shens, jpsweeney}@umich.edu

Abstract

This interactive presentation describes LexNet, a graphical environment for graph-based NLP developed at the University of Michigan. LexNet includes LexRank (for text summarization), biased LexRank (for passage retrieval), and TUMBL (for binary classification). All tools in the collection are based on random walks on *lexical graphs*, that is graphs where different NLP objects (e.g., sentences or phrases) are represented as nodes linked by edges proportional to the lexical similarity between the two nodes. We will demonstrate these tools on a variety of NLP tasks including summarization, question answering, and prepositional phrase attachment.

1 Introduction

We will present a series of graph-based tools for a variety of NLP tasks such as text summarization, passage retrieval, prepositional phrase attachment, and binary classification in general.

Recently proposed graph-based methods (Szummer and Jaakkola, 2001; Zhu and Ghahramani, 2002b; Zhu and Ghahramani, 2002a; Toutanova et al., 2004) are particularly well suited for transductive learning (Vapnik, 1998; Joachims, 1999). Transductive learning is based on the idea (Vapnik, 1998) that instead of splitting a learning problem into two possibly harder problems, namely induction and deduction, one can build a model that covers both labeled and unlabeled data. Unlabeled data are abundant as well as significantly cheaper than labeled data in a variety of natural language applications. Parsing and machine translation both offer examples of this relationship, with unparsed text from the Web and untranslated texts being computationally less costly. These can then be used to supplement manually translated and aligned corpora. Hence transductive methods are of great potential for NLP problems and, as a result, LexNet includes a number of transductive methods.

2 LexRank: text summarization

LexRank (Erkan and Radev, 2004) embodies the idea of representing a text (e.g., a document or a collection of related documents) as a graph. Each node corresponds to a sentence in the input and the edge between two nodes is related to the lexical similarity (either cosine similarity or n-gram generation probability) between the two sentences. LexRank computes the steady-state distribution of the random walk probabilities on this similarity graph. The LexRank score of each node gives the probability of a random walk ending up in that node in the long run. An extractive summary is generated by retrieving the sentences with the highest score in the graph. Such sentences typically correspond to the nodes that have strong connections to other nodes with high scores in the graph. Figure 1 demonstrates LexRank.

3 Biased LexRank: passage retrieval

The basic idea behind Biased LexRank is to label a small number of sentences (or passages) that are relevant to a particular query and then propagate relevance from these sentences to other (unannotated) sentences. Relevance propagation is performed on a bipartite graph. In that graph, one of the modes corresponds to the sentences and the other – to certain words from these sentences. Each sentence is connected to the words that appear in it. Thus indirectly, each sentence is two hops away from any other sentence that shares words in it. Intuitively, the sentences that are close to the labeled sentences tend to get higher scores. However, the relevance propagation en-

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 45–48,
Sydney, July 2006.

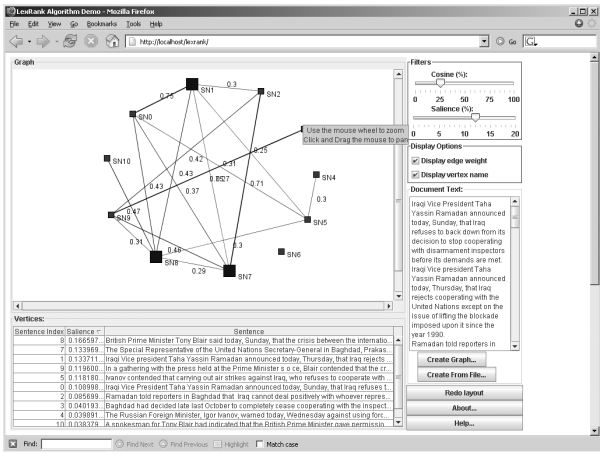


Figure 1: A sample snapshot of LexRank. A 3-sentence summary is produced from a set of 11 related input sentences. The summary sentences are shown as larger squares.

ables us to mark certain sentences that are not immediate neighbors of the labeled sentences via indirect connections. The effect of the propagation is discounted by a parameter at each step so that the relationships between closer nodes are favored more. Biased LexRank also allows for negative relevance to be propagated through the network as the example shows. See Figures 2– 3 for a demonstration of Biased LexRank.

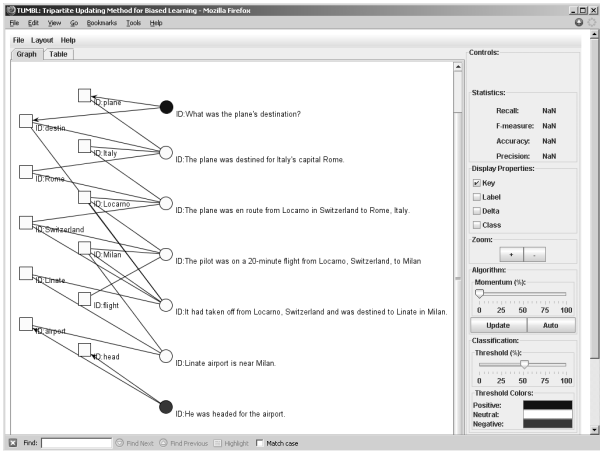


Figure 2: Display of Biased LexRank. One sentence at the top is annotated as positive while another at the bottom is marked negative. Sentences are displayed as circles and the word features are shown as squares.

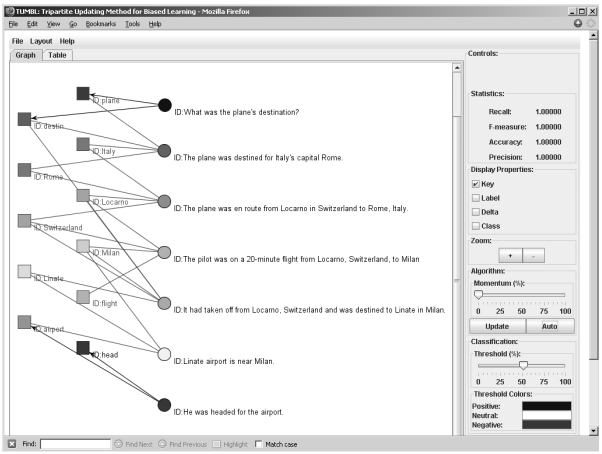


Figure 3: After convergence of Biased LexRank.

4 TUMBL: prepositional phrase attachment

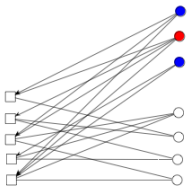
A number of NLP problems such as word sense disambiguation, text categorization, and extractive summarization can be cast as classification problems. This fact is used to great effect in the design and application of many machine learning methods used in modern NLP, including TUMBL, through the utilization of vector representations. Each object is represented as a vector $\mathbf{x}$ of features. The main assumption made is that a pair of objects $\mathbf{x}$ and $\mathbf{y}$ will be classified the same way if the distance between them in some space D is small (Zhu and Ghahramani, 2002a).

This algorithm propagates polarity information first from the labeled data to the features, capturing whether each feature is more indicative of positive class or more negative learned. Such information is further transferred to the unlabeled set. The backward steps update feature polarity with information learned from the structure of the unlabeled data. This process is repeated with a damping factor to discount later rounds. This process is illustracted in Figure 4. TUMBL was first described in (Radev, 2004). A series of snapshots showing TUMBL in Figures 5– 7.

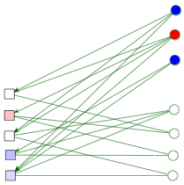
5 Technical information

5.1 Code implementation

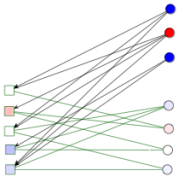
The LexRank and TUMBL demonstrations are provided as both an applet and an application. The user is presented with a graphical visualization of the algorithm that was conveniently developed using the JUNG API (`http://jung.sourceforge.net/faq.html`).



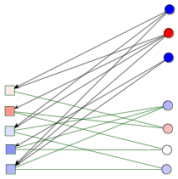
(a) Initial graph



(b) Forward pass



(c) Backward pass



(d) Convergence

Figure 4: TUMBL snapshots: the circular vertices are objects while the square vertices are features. (a) The initial graph with features showing no bias. (b) The forward pass where objects propagate labels forward. (c) The backward pass where features propagate labels backward. (d) Convergence of the TUMBL algorithm after successive iterations.

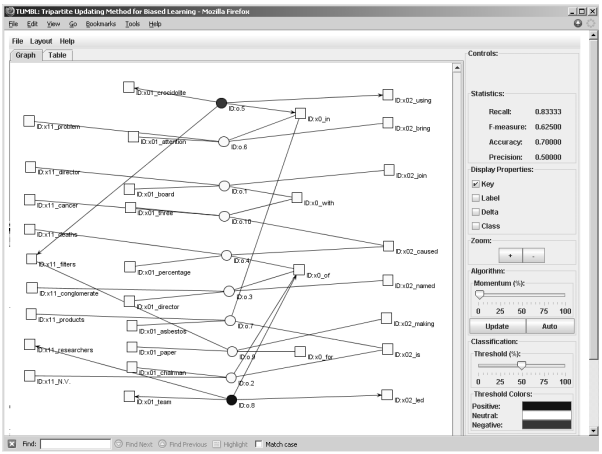


Figure 5: A 10-pp prepositional phrase attachment problem is displayed. The head of each prepositional phrase is ine middle column. Four types of features are represented in four columns. The first column is Noun1 in the 4-tuple. The second column is Noun2. The first column from the right is verb of the 4-tuple while the second column from the right is the actual head of the prepositional phrase. At this time one positive and one negative example (high and low attachment) are annotated. The rest of the circles correspond to the unlabeled examples.

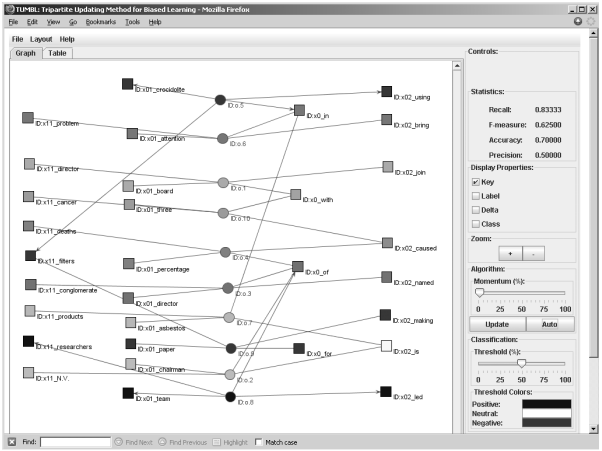


Figure 6: The final configuration.

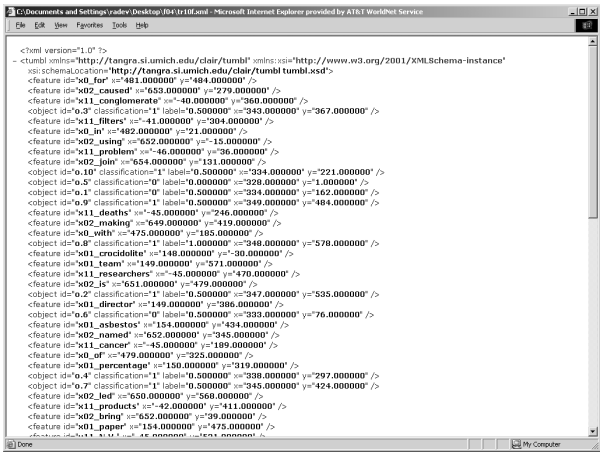


Figure 7: XML file corresponding to the PP attachment problem. The XML DTD allows layout information to be encoded along with algorithmic information such as label and polarity.

In TUMBL, each object is represented by a circular vertex in the graph and each feature as a square. Vertices are assigned a color according to their label. The colors are assignable by the user and designate the probability of membership of a class.

To allow for a range of uses, data can be entered either though the GUI or read in from an XML file. The schema for TUMBL files is shown at `http://tangra.si.umich.edu/clair/tumbl`.

In the LexRank demo, each sentence becomes a node. Selected nodes for the summary are shown in larger size and in blue while the rest are smaller and drawn in red. The link between two nodes has a weight proportional to the lexical similarity between the two corresponding sentences. The demo also reports the metrics precision, recall, and F-measure.

5.2 Availability

The demos are available both as locally based and remotely accessible from `http://tangra.si.umich.edu/clair/lexrank` and `http://tangra.si.umich.edu/clair/tumbl`.

6 Acknowledgments

This work was partially supported by the U.S. National Science Foundation under the following two grants: 0329043 “Probabilistic and link-based Methods for Exploiting Very Large Textual Repositories” administered through the IDM program and 0308024 “Collaborative Research: Semantic Entity and Relation Extraction from Web-Scale Text Document Collections” run by the HLC program. All opinions, findings, conclusions, and recommendations in this paper are made by the authors and do not necessarily reflect the views of the National Science Foundation.

References

Güneş Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR)*.

Thorsten Joachims. 1999. Transductive inference for text classification using support vector machines. In *ICML ’99*.

Dragomir Radev. 2004. Weakly supervised graph-based methods for classification. Technical Report CSE-TR-500-04, University of Michigan.

Martin Szummer and Tommi Jaakkola. 2001. Partially labeled classification with Markov random walks. In *NIPS ’01*, volume 14. MIT Pres.

Kristina Toutanova, Christopher D. Manning, and Andrew Y. Ng. 2004. Learning random walk models for inducing word dependency distributions. In *ICML ’04*, New York, New York, USA. ACM Press.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience.

Xiaojin Zhu and Zoubin Ghahramani. 2002a. Learning from labeled and unlabeled data with label propagation. Technical report, CMU-CALD-02-107.

Xiaojin Zhu and Zoubin Ghahramani. 2002b. Towards semi-supervised classification with Markov random fields. Technical report, CMU-CALD-02-106.

Archivus: A multimodal system for multimedia meeting browsing and retrieval

Marita Ailomaa, Miroslav Melichar, Martin Rajman
Artificial Intelligence Laboratory
École Polytechnique Fédérale de Lausanne
CH-1015 Lausanne, Switzerland
marita.ailomaa@epfl.ch

Agnes Lisowska, Susan Armstrong
ISSCO/TIM/ETI
University of Geneva
CH-1211 Geneva, Switzerland
agnes.lisowska@issco.unige.ch

Abstract

This paper presents Archivus, a multimodal language-enabled meeting browsing and retrieval system. The prototype is in an early stage of development, and we are currently exploring the role of natural language for interacting in this relatively unfamiliar and complex domain. We briefly describe the design and implementation status of the system, and then focus on how this system is used to elicit useful data for supporting hypotheses about multimodal interaction in the domain of meeting retrieval and for developing NLP modules for this specific domain.

1 Introduction

In the past few years, there has been an increasing interest in research on developing systems for efficient recording of and access to multimedia meeting data[1]. This work often results in videos of meetings, transcripts, electronic copies of documents referenced, as well as annotations of various kinds on this data. In order to exploit this work, a user needs to have an interface that allows them to retrieve and browse the multimedia meeting data easily and efficiently.

In our work we have developed a multimodal (voice, keyboard, mouse/pen) meeting browser, Archivus, whose purpose is to allow users to access multimedia meeting data in a way that is most natural to them. We believe that since this is a new domain of interaction, users can be encouraged to try out and consistently use novel input modalities such as voice, including more complex natural language, and that in particular in this domain, such multimodal interaction can help the user find information more efficiently.

When developing a language interface for an interactive system in a new domain, the Wizard of Oz (WOz) methodology (Dahlbäck et al., 1993; Salber and Coutaz, 1993) is a very useful tool. The user interacts with what they believe to be a fully automated system, when in fact another person, a 'wizard' is simulating the missing or incomplete NLP modules, typically the speech recognition, natural language understanding and dialogue management modules. The recorded experiments provide valuable information for implementing or fine-tuning these parts of the system.

However, the methodology is usually applied to unimodal (voice-only or keyboard-only) systems, where the elicitation of language data is not a problem since this is effectively the only type of data resulting from the experiment. In our case, we are developing a complex multimodal system. We found that when the Wizard of Oz methodology is extended to multimodal systems, the number of variables that have to be considered and controlled for in the experiment increases substantially. For instance, if it is the case that within a single interface any task that can be performed using natural language can also be performed with other modalities, for example a mouse, the user may prefer to use the other – more familiar – modality for a sizeable portion of the experiment. In order to gather a useful amount of natural language data, greater care has to be taken to design the system in a way that encourages language use. But, if the goal of the experiment is also to study what modalities users find more useful in some situa-

[1]The IM2 project *http://www.im2.ch*, the AMI project *www.amiproject.org*, The Meeting Room Project at Carnegie Mellon University, *http://www.is.cs.cmu.edu/mie*, and rich transcription of natural and impromptu meetings at ICSI, Berkeley, *http://www.icsi.berkeley.edu/Speech/EARS/rt.html*

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 49–52,
Sydney, July 2006.

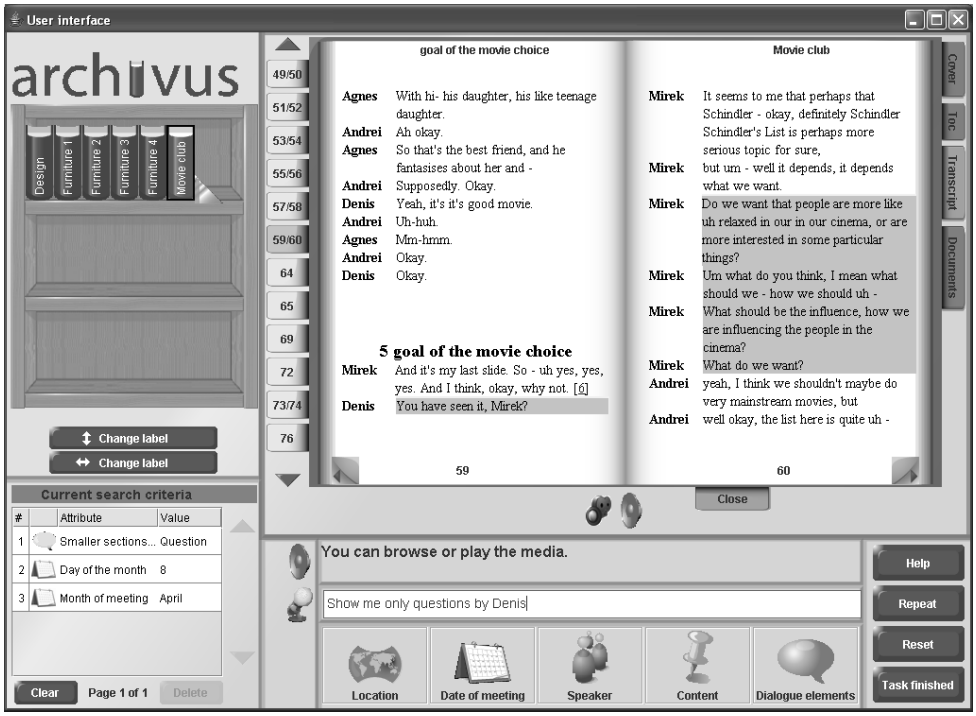


Figure 1: The Archivus Interface

tions compared to others, language use must be encouraged without being forced, and finding this balance can be very hard to achieve in practice.

2 Design and implementation

The Archivus system has been designed to satisfy realistic user needs based on a user requirement analysis (Lisowska, 2003), where subjects were asked to formulate queries that would enable them to find out "what happened at a meeting". The design of the user interface is based on the metaphor of a person interacting in an archive or library (Lisowska et al., 2004).

Furthermore, Archivus is flexibly multimodal, meaning that users can interact unimodally choosing one of the available modalities exclusively, or multimodally, using any combination of the modalities. In order to encourage natural language interaction, the system gives textual and vocal feedback to the user. The Archivus Interface is shown in Figure 1. A detailed description of all of the components can be found in Lisowska et al. (2004).

Archivus was implemented within a software framework for designing multimodal applications with mixed-initiative dialogue models (Cenek et al., 2005). Systems designed within this framework handle interaction with the user through a multimodal dialogue manager. The dialogue manager receives user input from all modalities (speech, typing and pointing) and provides multimodal responses in the form of graphical, textual and vocal feedback.

The dialogue manager contains only linguistic knowledge and interaction algorithms. Domain knowledge is stored in an SQL database and is accessed by the dialogue manager based on the constraints expressed by the user during interaction.

The above software framework provides support for remote simulation or supervision of some of the application functionalities. This feature makes any application developed under this methodology well suited for WOz experiments. In the case of Archivus, pilot experiments strongly suggested the use of two wizards – one supervising the user's input (Input Wizard) and the other controlling the natural language output of the system (Output Wizard). Both wizards see the user's input, but their actions are sequential, with the Output Wizard being constrained by the actions of the Input Wizard.

The role of the Input Wizard is to assure that the user's input (in any modality combination) is correctly conveyed to the system in the form of sets of semantic pairs. A semantic pair (SP) is a qualified piece of information that the dia-

logue system is able to understand. For example, a system could understand semantic pairs such as `date:Monday` or `list:next`. A user's utterance *"What questions did this guy ask in the meeting yesterday?"* combined with pointing on the screen at a person called "Raymond" could translate to `dialogact:Question`, `speaker:Raymond`, `day:Monday`.

In the current version of Archivus, user clicks are translated into semantic pairs automatically by the system. Where written queries are concerned, the wizard sometimes needs to correct automatically generated pairs due to the currently low performance of our natural language understanding module. Finally since the speech recognition engine has not been implemented yet, the user's speech is fully processed by a wizard. The Input Wizard also assures that the fusion of pairs coming from different modalities is done correctly.

The role of the Output Wizard is to monitor, and if necessary change the default prompts that are generated by the system. Changes are made for example to smooth the dialogue flow, i.e. to better explain the dialogue situation to the user or to make the response more conversational. The wizard can select a prompt from a predefined list, or type a new one during interaction.

All wizards' actions are logged and afterwards used to help automate the correct behavior of the system and to increase the overall performance.

3 Collecting natural language data

In order to obtain a sufficient amount of language data from the WOz experiments, several means have been used to determine what encourages users to speak to the system. These include giving users different types of documentation before the experiment – lists of possible voice commands, a user manual, and step-by-step tutorials. We found that the best solution was to give users a tutorial in which they worked through an example using voice alone or in combination with other modalities, explaining in each step the consequences of the user's actions on the system. The drawback of this approach is that the user may be biased by the examples and continue to interact according to the interaction patterns that are provided, rather than developing their own patterns. These influences need to be considered both in the data analysis, and in how the tutorials are written and structured.

The actual experiment consists of two parts in which the user gets a mixed set of short-answer and true-false questions to solve using the system. First they are only allowed to use a subset of the available modalities, e.g. voice and pen, and then the full set of modalities. By giving the users different subsets in the first part, we can compare if the enforcement of certain modalities has an impact on how they choose to use language when all modalities are available.

On the backend, the wizards can also to some extent have an active role in encouraging language use. The Input Wizard is rather constrained in terms of what semantic pairs he can produce, because he is committed to selecting from a set of pairs that are extracted from the meeting data. For example if "Monday" is not a meeting date in the database, the input is interpreted as having "no match", which generates the system prompt *"I don't understand"*. Here, the Output Wizard can intervene by replacing that prompt by one that more precisely specifies the nature of the problem.

The Output Wizard can also decide to replace default prompts in situations when they are too general in a given context. For instance, when the user is browsing different sections of a meeting book (cover page, table of contents, transcript and referenced documents) the default prompt gives general advice on how to access the different parts of the book, but can be changed to suggest a specific section instead.

4 Analysis of elicited language data

The data collected with Archivus through WOz experiments provide useful information in several ways. One aspect is to see the complexity of the language used by users – for instance whether they use more keywords, multi-word expressions or full-sentence queries. This is important for choosing the appropriate level of language processing, for instance for the syntactic analysis. Another aspect is to see the types of actions performed using language. On one hand, users can manipulate elements in the graphical interface by expressing commands that are semantically equivalent with pointing, e.g. *"next page"*. On the other hand, they can freely formulate queries relating to the information they are looking for, e.g. *"Did they decide to put a sofa in the lounge?"*. Commands are interface specific rather than domain specific. From the graphical interface the user can easily predict what they can say and how the system will

Part 1 condition	Pointing	Language
Experiment set 1		
voice only	91%	9%
voice+keyboard	88%	12%
keyboard+pointing	66%	34%
voice+keyb.+pointing	79%	21%
Experiment set 2		
voice only	68%	32%
voice+pointing	62%	38%
keyboard+pointing	39%	61%
pointing	76%	24%

Table 1: Use of each modality in part 2.

respond. Queries depend on the domain and the data, and are more problematic for the user because they cannot immediately see what types of queries they can ask and what the coverage of the data is. But, using queries can be very useful, because it allows the user to express themselves in their own terms. An important goal of the data analysis is to determine if the language interface enables the user to interact more successfully than if they are limited to pointing only. In addition, the way in which the users use language in these two dimensions has important implications for the dialogue strategy and for the implementation of the language processing modules, for instance the speech recognition engine. A speech recognizer can be very accurate when trained on a small, fixed set of commands whereas it may perform poorly when faced with a wide variety of language queries.

Thus far, we have performed 3 sets of pilot WOz experiments with 40 participants. The primary aim was to improve and finetune the system and the WOz environment as a preparation for the data-collection experiments that we plan to do in the future. In these experiments we compared how frequently users used voice and keyboard in relation to pointing as we progressively changed features in the system and the experimental setup to encourage language use. The results between the first and the third set of experiments can be seen in table 1, grouped by the subset of modalities that the users had in the first part of the experiment.

From the table we can see that changes made between the different iterations of the system achieved their goal – by the third experiment set we were managing to elicit larger amounts of natural language data. Moreover, we noticed that the modality conditions that are available to the user in the first part play a role in the amount of use of language modalities in the second part.

5 Conclusions and future work

We believe that the work presented here (both the system and the WOz environment and experimental protocol) has now reached a stable stage that allows for the elicitation of sufficient amounts of natural language and interaction data. The next step will be to run a large-scale data collection. The results from this collection should provide enough information to allow us to develop and integrate fairly robust natural language processing into the system. Ideally, some of the components used in the software framework will be made publicly available at the end of the project.

References

Pavel Cenek, Miroslav Melichar, and Martin Rajman. 2005. A Framework for Rapid Multimodal Application Design. In Václav Matoušek, Pavel Mautner, and Tomáš Pavelka, editors, *Proceedings of the 8th International Conference on Text, Speech and Dialogue (TSD 2005)*, volume 3658 of *Lecture Notes in Computer Science*, pages 393–403, Karlovy Vary, Czech Republic, September 12-15. Springer.

Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz Studies – Why and How. In Dianne Murray Wayne D. Gray, William Hefley, editor, *International Workshop on Intelligent User Interfaces 1993*, pages 193–200. ACM Press.

Agnes Lisowska, Martin Rajman, and Trung H. Bui. 2004. ARCHIVUS: A System for Accessing the Content of Recorded Multimodal Meetings. In *In Procedings of the JOINT AMI/PASCAL/IM2/M4 Workshop on Multimodal Interaction and Related Machine Learning Algorithms, Bourlard H. & Bengio S., eds. (2004), LNCS, Springer-Verlag, Berlin.*, Martigny, Switzerland, June.

Agnes Lisowska. 2003. Multimodal interface design for the multimodal meeting domain: Preliminary indications from a query analysis study. Project report IM2.MDM-11, University of Geneva, Geneva, Switzerland, November.

Daniel Salber and Joëlle Coutaz. 1993. Applying the wizard of oz technique to the study of multimodal systems. In *EWHCI '93: Selected papers from the Third International Conference on Human-Computer Interaction*, pages 219–230, London, UK. Springer-Verlag.

Re-Usable Tools for Precision Machine Translation*

Jan Tore Lønning♣ and Stephan Oepen♣♠

♣Universitetet i Oslo, Computer Science Institute, Boks 1080 Blindern; 0316 Oslo (Norway)
♠Center for the Study of Language and Information, Stanford, CA 94305 (USA)
{ jtl@ifi.uio.no|oe@csli.stanford.edu}

Abstract

The LOGON MT demonstrator assembles independently valuable general-purpose NLP components into a machine translation pipeline that capitalizes on output quality. The demonstrator embodies an interesting combination of hand-built, symbolic resources and stochastic processes.

1 Background

The LOGON projects aims at building an experimental machine translation system from Norwegian to English of texts in the domain of hiking in the wilderness (Oepen et al., 2004). It is funded within the Norwegian Research Council program for building national infrastructure for language technology (Fenstad et al., 2006). It is the goal for the program as well as for the project to include various areas of language technology as well as various methods, in particular symbolic and empirical methods. Besides, the project aims at reusing available resources and, in turn, producing re-usable technology.

In spite of significant progress in statistical approaches to machine translation, we doubt the long-term value of *pure* statistical (or data-driven) approaches, both practically and scientifically. To ensure grammaticality of outputs as well as felicity of the translation both linguistic grammars and deep semantic analysis are needed. The architecture of the LOGON system hence consists of a symbolic backbone system combined with various stochastic components for ranking system hypotheses. In a nutshell, a central research question in LOGON is to what degree state-of-the-art 'deep' NLP resources can contribute towards a precision MT system. We hope to engage the conference audience in some reflection on this question by means of the interactive presentation.

2 System Design

The backbone of the LOGON prototype implements a relatively conventional architecture, organized around in-depth grammatical analysis in the source language (SL), semantic transfer of logical-form meaning representations from the source into the target language (TL), and full, grammar-based TL tactical generation.

$\langle h_1,$
$\{ h_1$:proposition_m(h_3),
h_4:proper_q(x_5, h_6, h_7), h_8:named(x_5,'Bodø'),
h_9:_populate_v(e_2, _, x_5), h_9:_densely_r(e_2) },
$\{ h_3 =_q h_9, h_6 =_q h_8 \} \rangle$

Figure 1: Simplified MRS representation for the utterance 'Bodø is densely populated.' The core of the structure is a bag of *elementary predications* (EPs), using distinguished handles ('h_i' variables) and '$=_q$' (equal modulo quantifier insertion) constraints to underspecify scopal relations. Event- and instance-type variables ('e_j' and 'x_k', respectively) capture semantic linking among EPs, where we assume a small inventory of thematically bleached role labels (ARG_0 ... ARG_n). These are abbreviated through order-coding in the example above (see § 2 below for details).

Minimal Recursion Semantics The three core phases communicate in a uniform semantic interface language, Minimal Recursion Semantics (MRS; Copestake, Flickinger, Sag, & Pollard, 1999). Broadly speaking, MRS is a flat, event-based (neo-Davidsonian) framework for computational semantics. The abstraction from SL and TL surface properties enforced in our semantic transfer approach facilitates a novel combination of diverse grammatical frameworks, viz. LFG for Norwegian analysis and HPSG for English generation.

While an in-depth introduction to MRS (for MT) is beyond the scope of this project note, Figure 1 presents a simplified example semantics.

Norwegian Analysis Syntactic analysis of Norwegian is based on an existing LFG resource grammar, NorGram (Dyvik, 1999), under development on the Xerox Linguistic Environment (XLE) since around 1999. For use in LOGON, the grammar has been modified and extended, and it has been augmented with a module of Minimal Recursion Semantics representations which are computed from LFG f-structures by co-description.

In Norwegian, compounding is a productive morphological process, thus presenting the analysis engine with a steady supply of 'new' words, e.g. something like *klokkeslettuttrykk* meaning ap-

This demonstration reflects the work of a large group of people whose contributions we gratefully acknowledge. Please see 'http://www.emmtee.net' for background.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 53–56,
Sydney, July 2006.

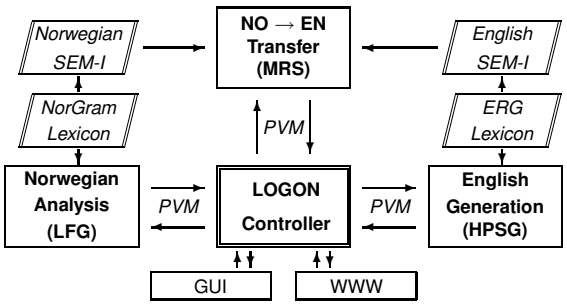


Figure 2: Schematic system architecture: the three core processing components are managed by a central controller that passes intermediate results (MRSs) through the translation pipeline. The Parallel Virtual Machine (PVM) layer facilitates distribution, parallelization, failure detection, and roll-over.

'lingo/jan-06/jh1/06-01-20/lkb' Generation Profile					
Aggregate	**total items** ♯	**word string** ϕ	**distinct trees** ϕ	**overall coverage** %	**time (s)** ϕ
$30 \leq$ *i-length* < 40	21	33.1	241.5	61.9	36.5
$20 \leq$ *i-length* < 30	174	23.0	158.6	80.5	15.7
$10 \leq$ *i-length* < 20	353	14.3	66.7	86.7	4.1
$0 \leq$ *i-length* < 10	495	4.6	6.0	90.1	0.7
Total	**1044**	**11.6**	**53.50**	**86.7**	**4.3**

(generated by [incr tsdb()] at 15-mar-2006 (15:51 h))

Table 1: Central measures of generator performance in relation to input 'complexity'. The columns are, from left to right, the corpus sub-division by input length, total number of items, and average string length, ambiguity rate, grammatical coverage, and generation time, respectively.

proximately *time-of-day expression*. The project uses its own morphological analyzer, compiled off a comprehesive computational lexicon of Norwegian, prior to syntactic analysis. One important feature of this processor is that it decomposes compounds in such a way that they can be compositionally translated downstream.

Current analysis coverage (including well-formed MRSs) on the LOGON corpus (see below) is approaching 80 per cent (of which 25 per cent are 'fragmented', i.e. approximative analyses).

Semantic Transfer Unlike in parsing and generation, there is less established common wisdom in terms of (semantic) transfer formalisms and algorithms. LOGON follows many of the main Verb*mobil* ideas—transfer as a resource-sensitive rewrite process, where rules replace MRS fragments (SL to TL) in a step-wise manner (Wahlster, 2000)—but adds two innovative elements to the transfer component, viz. (i) the use of typing for hierarchical organization of transfer rules and (ii) a chart-like treatment of transfer-level ambiguity. The general form of MRS transfer rules (MTRs) is as a quadruple:

[CONTEXT :] INPUT [! FILTER] → OUTPUT

where each of the four components, in turn, is a partial MRS, i.e. triplet of a top handle, bag of EPs, and handle constraints. Left-hand side components are unified against an input MRS M and, when successful, trigger the rule application; elements of M matched by INPUT are replaced with the OUTPUT component, respecting all variable bindings established during unification. The optional CONTEXT and FILTER components serve to condition rule application (on the presence or absence of specific aspects of M), establish bindings for OUTPUT processing, but do *not* consume elements of M. Although our current focus is on translation into English, MTRs in principle state translational correspondence relations and, modulo context conditioning, can be reversed.

Transfer rules use a multiple-inheritance hierarchy with strong typing and appropriate feature constraints both for elements of MRSs and MTRs themselves. In close analogy to constraint-based grammar, typing facilitates generalizations over transfer regularities—hierarchies of predicates or common MTR configurations, for example—and aids development and debugging.

An important tool in the constructions of the transfer rules are the semantic interfaces (called SEM-Is, see below) of the respective grammars. While we believe that hand-crafted lexical transfer is a necessary component in precision-oriented MT, it is also a bottleneck for the development of the LOGON system, with its pre-existing source and target language grammars. We have therefore experimented with the acquistion of transfer rules by analogy from a bi-lingual dictionary, building on hand-built transfer rules as a seed set of templates (Nordgård, Nygaard, Lønning, & Oepen, 2006).

English Generation Realization of post-transfer MRSs in LOGON builds on the pre-existing LinGO English Resource Grammar (ERG; Flickinger, 2000) and LKB generator (Carroll, Copestake, Flickinger, & Poznanski, 1999). The ERG already produced MRS outputs with good coverage in several domains. In LOGON, it has been refined, adopted to the new domain, and semantic representations revised in light of cross-linguistic experiences from MT. Furthermore, chart generation efficiency and integration with stochastic realization have been substantially improved (Carroll & Oepen, 2005). Table 1 summarizes (exhaustive) generator performance on a segment of the LOGON

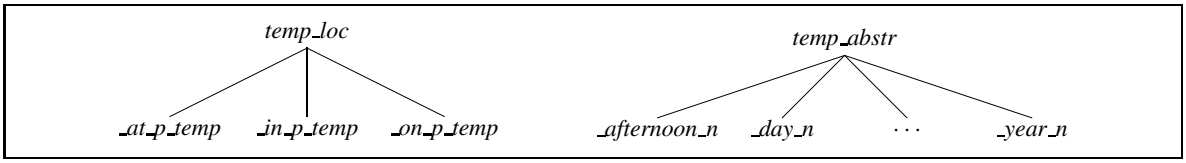


Figure 3: Excerpt from predicate hierarchies provided by English SEM-I. Temporal, directional, and other usages of prepositions give rise to distinct, but potentially related, semantic predicates. Likewise, the SEM-I incorporates some ontological information, e.g. a classification of temporal entities, though crucially only to the extent that is actually grammaticized in the language proper.

development corpus: realizations average at a little less than twelve words in length. After addition of domain-specific vocabulary and a small amount of fine-tuning, the ERG provides adequate analyses for close to ninety per cent of the LOGON reference translations. For about half the test cases, all outputs can be generated in less than one cpu second.

End-to-End Coverage The current LOGON system will only produce output(s) when all three processing phases succeed. For the LOGON target corpus (see below), this is presently the case in 35 per cent of cases. Averaging over actual outputs only, the system achieves a (respectable) BLEU score of 0.61; averaging over the entire corpus, i.e. counting inputs with processing errors as a zero contribution, the BLEU score drops to 0.21.

3 Stochastic Components

To deal with competing hypotheses at all processing levels, LOGON incorporates various stochastic processes for disambiguation. In the following, we present the ones that are best developed to date.

Training Material A corpus of some 50,000 words of edited, running Norwegian text was gathered and translated by three professional translators. Three quarters of the material are available for system development and also serve as training data for machine learning approaches. Using the discriminant-based Redwoods approach to treebanking (Oepen, Flickinger, Toutanova, & Manning, 2004), a first 5,000 English reference translations were hand-annotated and released to the public.[1] In on-going work on adapting the Redwoods approach to (Norwegian) LFG, we are working to treebank a sizable text segment (Rosén, Smedt, Dyvik, & Meurer, 2005; Oepen & Lønning, 2006).

Parse Selection The XLE analyzer includes support for stochastic parse selection models, assigning likelihood measures to competing analyses (Riezler et al., 2002). Using a trial LFG treebank for Norwegian (of less than 100 annotated sentences), we have adapted the tools for the current LOGON version and are now working to train on larger data sets and evaluate parse selection performance. Despite the very limited amount of training so far, the model already appears to pick up on plausible, albeit crude preferences (as regards topicalization, for example). Furthermore, to reduce fan-out in exhaustive processing, we collapse analyses that project equivalent MRSs, i.e. syntactic distinctions made in the grammar but not reflected in the semantics.

Realization Ranking At an average of more than fifty English realizations per input MRS (see Table 1), ranking generator outputs is a vital part of the LOGON pipeline. Based on a notion of automatically derived *symmetric treebanks*, we have trained comprehensive discriminative, log-linear models that (within the LOGON domain) achieve up to 75 per cent exact match accuracy in picking the most likely realization among competing outputs (Velldal & Oepen, 2005). The best-performing models make use of configurational (in terms of tree topology) as well as of string-level properties (including local word order and constituent weight), both with varied domains of locality. In total, there are around 300,000 features with non-trivial distribution, and we combine the MaxEnt model with a traditional language model trained on a much larger corpus (the BNC). The latter, more standard approach to realization ranking, when used in isolation only achieves around 50 per cent accuracy, however.

4 Implementation

Figure 2 presents the main components of the LOGON prototype, where all component communication is in terms of sets of MRSs and, thus, can easily be managed in a distributed and (potentially) parallel client – server set-up. Both the analysis and generation grammars 'publish' their interface to transfer—i.e. the inventory and synopsis of seman-

[1] See 'http://www.delph-in.net/redwoods/' for the LinGO Redwoods treebank in its latest release, dubbed Norwegian Growth.

tic predicates—in the form of a Semantic Interface specification ('SEM-I'; Flickinger, Lønning, Dyvik, Oepen, & Bond, 2005), such that transfer can operate without knowledge about grammar internals. In practical terms, SEM-Is are an important development tool (facilitating well-formedness testing of interface representations at all levels), but they also have interesting theoretical status with regard to transfer. The SEM-Is for the Norwegian analysis and English generation grammars, respectively, provide an exhaustive enumeration of legitimate semantic predicates (i.e. the transfer vocabulary) and 'terms of use', i.e. for each predicate its set of appropriate roles, corresponding value constraints, and indication of (semantic) optionality of roles. Furthermore, the SEM-I provides generalizations over classes of predicates—e.g. hierarchical relations like those depicted in Figure 3 below—that play an important role in the organization of MRS transfer rules.

5 Open-Source Machine Translation

Despite the recognized need for translation, there is no widely used open-source machine translation system. One of the major reasons for this lack of success is the complexity of the task. By association to the international open-source DELPH-IN effort[2] and with its strong emphasis on re-usability, LOGON aims to help build a repository of open-source precision tools. This means that work on the MT system benefits other projects, and work on other projects can improve the MT system (where EBMT and SMT systems provide results that are harder to re-use). While the XLE software used for Norwegian analysis remains proprietary, we have built an open-source bi-directional Japanese – English prototype adaptation of the LOGON system (Bond, Oepen, Siegel, Copestake, & Flickinger, 2005). This system will be available for public download by the summer of 2006.

References

Bond, F., Oepen, S., Siegel, M., Copestake, A., & Flickinger, D. (2005). Open source machine translation with DELPH-IN. In *Proceedings of the Open-Source Machine Translation workshop at the 10th Machine Translation Summit* (pp. 15 – 22). Phuket, Thailand.

Carroll, J., Copestake, A., Flickinger, D., & Poznanski, V. (1999). An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation* (pp. 86 – 95). Toulouse, France.

Carroll, J., & Oepen, S. (2005). High-efficiency realization for a wide-coverage unification grammar. In R. Dale & K.-F. Wong (Eds.), *Proceedings of the 2nd International Joint Conference on Natural Language Processing* (Vol. 3651, pp. 165 – 176). Jeju, Korea: Springer.

Copestake, A., Flickinger, D., Sag, I. A., & Pollard, C. (1999). *Minimal Recursion Semantics. An introduction.* In preparation, CSLI Stanford, Stanford, CA.

Dyvik, H. (1999). The universality of f-structure. Discovery or stipulation? The case of modals. In *Proceedings of the 4th International Lexical Functional Grammar Conference.* Manchester, UK.

Fenstad, J.-E., Ahrenberg, L., Kvale, K., Maegaard, B., Mühlenbock, K., & Heid, B.-E. (2006). KUNSTI. Knowledge generation for Norwegian language technology. In *Proceedings of the 5th International Conference on Language Resources and Evaluation.* Genoa, Italy.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, *6 (1)*, 15 – 28.

Flickinger, D., Lønning, J. T., Dyvik, H., Oepen, S., & Bond, F. (2005). SEM-I rational MT. Enriching deep grammars with a semantic interface for scalable machine translation. In *Proceedings of the 10th Machine Translation Summit* (pp. 165 – 172). Phuket, Thailand.

Nordgård, T., Nygaard, L., Lønning, J. T., & Oepen, S. (2006). Using a bi-lingual dictionary in lexical transfer. In *Proceedings of the 11th conference of the European Asoociation of Machine Translation.* Oslo, Norway.

Oepen, S., Dyvik, H., Lønning, J. T., Velldal, E., Beermann, D., Carroll, J., Flickinger, D., Hellan, L., Johannessen, J. B., Meurer, P., Nordgård, T., & Rosén, V. (2004). Som å kapp-ete med trollet? Towards MRS-based Norwegian – English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation.* Baltimore, MD.

Oepen, S., Flickinger, D., Toutanova, K., & Manning, C. D. (2004). LinGO Redwoods. A rich and dynamic treebank for HPSG. *Journal of Research on Language and Computation*, *2*(4), 575 – 596.

Oepen, S., & Lønning, J. T. (2006). Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation.* Genoa, Italy.

Riezler, S., King, T. H., Kaplan, R. M., Crouch, R., Maxwell, J. T., & Johnson, M. (2002). Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics.* Philadelphia, PA.

Rosén, V., Smedt, K. D., Dyvik, H., & Meurer, P. (2005). TrePil. Developing methods and tools for multilevel treebank construction. In *Proceedings of the 4th Workshop on Treebanks and Linguistic Theories* (pp. 161 – 172). Barcelona, Spain.

Velldal, E., & Oepen, S. (2005). Maximum entropy models for realization ranking. In *Proceedings of the 10th Machine Translation Summit* (pp. 109 – 116). Phuket, Thailand.

Wahlster, W. (Ed.). (2000). *Verbmobil. Foundations of speech-to-speech translation.* Berlin, Germany: Springer.

[2] See 'http://www.delph-in.net' for details, including the lists of participating sites and already available resources.

The SAMMIE System: Multimodal In-Car Dialogue

Tilman Becker, Peter Poller, Jan Schehl
DFKI
First.Last@dfki.de

Nate Blaylock, Ciprian Gerstenberger, Ivana Kruijff-Korbayová
Saarland University
talk-mit@coli.uni-sb.de

Abstract

The SAMMIE[1] system is an in-car multimodal dialogue system for an MP3 application. It is used as a testing environment for our research in natural, intuitive mixed-initiative interaction, with particular emphasis on multimodal output planning and realization aimed to produce output adapted to the context, including the driver's attention state w.r.t. the primary driving task.

1 Introduction

The SAMMIE system, developed in the TALK project in cooperation between several academic and industrial partners, employs the Information State Update paradigm, extended to model collaborative problem solving, multimodal context and the driver's attention state. We performed extensive user studies in a WOZ setup to guide the system design. A formal usability evaluation of the system's baseline version in a laboratory environment has been carried out with overall positive results. An enhanced version of the system will be integrated and evaluated in a research car.

In the following sections, we describe the functionality and architecture of the system, point out its special features in comparison to existing work, and give more details on the modules that are in the focus of our research interests. Finally, we summarize our experiments and evaluation results.

2 Functionality

The SAMMIE system provides a multi-modal interface to an in-car MP3 player (see Fig. 1) through speech and haptic input with a BMW iDrive input device, a button which can be turned, pushed down and sideways in four directions (see Fig. 2 left). System output is provided by speech and a graphical display integrated into the car's dashboard. An example of the system display is shown in Fig. 2.

[1] SAMMIE stands for Saarbrücken Multimodal MP3 Player Interaction Experiment.

Figure 1: User environment in laboratory setup.

The MP3 player application offers a wide range of functions: The user can control the currently playing song, search and browse an MP3 database by looking for any of the fields (song, artist, album, year, etc.), search and select playlists and even construct and edit playlists.

The user of SAMMIE has complete freedom in interacting with the system. Input can be through any modality and is not restricted to answers to system queries. On the contrary, the user can give new tasks as well as any information relevant to the current task at any time. This is achieved by modeling the interaction as a collaborative problem solving process, and multi-modal interpretation that fits user input into the context of the current task. The user is also free in their use of multimodality: SAMMIE handles deictic references (e.g., *Play this title* while pushing the iDrive button) and also cross-modal references, e.g., *Play the third song (on the list)*. Table 1 shows a typical interaction with the SAMMIE system; the displayed song list is in Fig. 2. SAMMIE supports interaction in German and English.

3 Architecture

Our system architecture follows the classical approach (Bunt et al., 2005) of a pipelined architecture with multimodal interpretation (fusion) and

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 57–60,
Sydney, July 2006.

U:	Show me the Beatles albums.
S:	I have these four Beatles albums. [shows a list of album names]
U:	Which songs are on this one? [selects the Red Album]
S:	The Red Album contains these songs [shows a list of the songs]
U:	Play the third one.
S:	[music plays]

Table 1: A typical interaction with SAMMIE.

fission modules encapsulating the dialogue manager. Fig. 2 shows the modules and their interaction: Modality-specific recognizers and analyzers provide semantically interpreted input to the multimodal fusion module that interprets them in the context of the other modalities and the current dialogue context. The dialogue manager decides on the next system move, based on its model of the tasks as collaborative problem solving, the current context and also the results from calls to the MP3 database. The turn planning module then determines an appropriate message to the user by planning the content, distributing it over the available output modalities and finally co-ordinating and synchronizing the output. Modality-specific output modules generate spoken output and graphical display update. All modules interact with the extended information state which stores all context information.

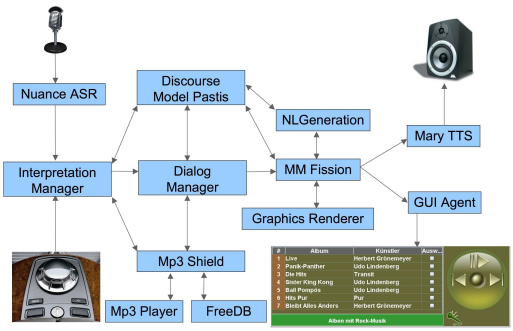


Figure 2: SAMMIE system architecture.

Many tasks in the SAMMIE system are modeled by a plan-based approach. Discourse modeling, interpretation management, dialogue management and linguistic planning, and turn planning are all based on the production rule system PATE[2] (Pfleger, 2004). It is based on some concepts of the ACT-R 4.0 system, in particular the goal-oriented application of production rules, the activation of working memory elements, and the weighting of production rules. In processing typed feature structures, PATE provides two operations that both integrate data and also are suitable for condition matching in production rule systems, namely a slightly extended version of the general *unification*, but also the discourse-oriented operation *overlay* (Alexandersson and Becker, 2001).

4 Related Work and Novel Aspects

Many dialogue systems deployed today follow a state-based approach that explicitly models the full (finite) set of dialogue states and all possible transitions between them. The VoiceXML[3] standard is a prominent example of this approach. This has two drawbacks: on the one hand, this approach is not very flexible and typically allows only so-called system controlled dialogues where the user is restricted to choosing their input from provided menu-like lists and answering specific questions. The user never is in control of the dialogue. For restricted tasks with a clear structure, such an approach is often sufficient and has been applied successfully. On the other hand, building such applications requires a fully specified model of all possible states and transitions, making larger applications expensive to build and difficult to test.

In SAMMIE we adopt an approach that models the interaction on an abstract level as collaborative problem solving and adds application specific knowledge on the possible *tasks*, available *resources* and known *recipes* for achieving the goals.

In addition, all relevant context information is administered in an Extended Information State. This is an extension of the Information State Update approach (Traum and Larsson, 2003) to the multi-modal setting.

Novel aspects in turn planning and realization include the comprehensive modeling in a single, OWL-based ontology and an extended range of context-sensitive variation, including system alignment to the user on multiple levels.

5 Flexible Multi-modal Interaction

5.1 Extended Information State

The information state of a multimodal system needs to contain a representation of contextual information about discourse, but also a representation of modality-specific information and user-specific information which can be used to plan system output suited to a given context. The over-

[2] Short for (P)roduction rule system based on (A)ctivation and (T)yped feature structure (E)lements.

[3] `http://www.w3.org/TR/voicexml20`

all information state (IS) of the SAMMIE system is shown in Fig. 3.

The contextual information partition of the IS represents the multimodal discourse context. It contains a record of the latest user utterance and preceding discourse history representing in a uniform way the salient discourse entities introduced in the different modalities. We adopt the three-tiered multimodal context representation used in the SmartKom system (Pfleger et al., 2003). The contents of the task partition are explained in the next section.

5.2 Collaborative Problem Solving

Our dialogue manager is based on an agent-based model which views dialogue as collaborative problem-solving (CPS) (Blaylock and Allen, 2005). The basic building blocks of the formal CPS model are problem-solving (PS) objects, which we represent as typed feature structures. PS object types form a single-inheritance hierarchy. In our CPS model, we define types for the upper level of an ontology of PS objects, which we term *abstract PS objects*. There are six abstract PS objects in our model from which all other domain-specific PS objects inherit: objective, recipe, constraint, evaluation, situation, and resource. These are used to model problem-solving at a domain-independent level and are taken as arguments by all update operators of the dialogue manager which implement conversation acts (Blaylock and Allen, 2005). The model is then specialized to a domain by inheriting and instantiating domain-specific types and instances of the PS objects.

5.3 Adaptive Turn Planning

The *fission* component comprises detailed content planning, media allocation and coordination and synchronization. Turn planning takes a set of CPS-specific conversational acts generated by the dialogue manager and maps them to modality-specific communicative acts.

Information on how content should be distributed over the available modalities (speech or graphics) is obtained from *Pastis*, a module which stores discourse-specific information. Pastis provides information about (i) the modality on which the user is currently focused, derived by the current discourse context; (ii) the user's current cognitive load when system interaction becomes a secondary task (e.g., system interaction while driving); (iii) the user's expertise, which is represented as a state variable. Pastis also contains information about factors that influence the preparation of output rendering for a modality, like the currently used language (German or English) or the display capabilities (e.g., maximum number of displayable objects within a table). Together with the dialogue manager's embedded part of the information state, the information stored by Pastis forms the *Extended Information State* of the SAMMIE system (Fig. 3).

Planning is then executed through a set of production rules that determine which kind of information should be presented through which of the available modalities. The rule set is divided in two subsets, domain-specific and domain-independent rules which together form the system's multimodal plan library.

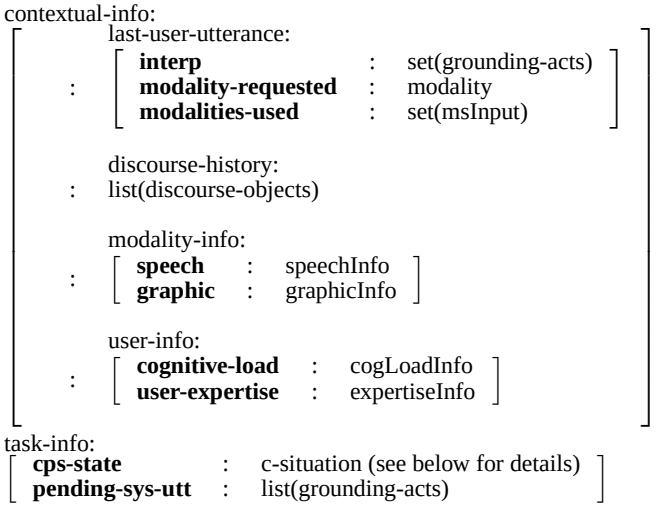


Figure 3: SAMMIE Information State structure.

5.4 Spoken Natural Language Output Generation

Our goal is to produce output that varies in the surface realization form and is adapted to the context. A template-based module has been developed and is sufficient for classes of system output that do not need fine-tuned context-driven variation. Our template-based generator can also deliver alternative realizations, e.g., alternative syntactic constructions, referring expressions, or lexical items. It is implemented by a set of straightforward sentence planning rules in the PATE system to build the templates, and a set of XSLT transformations to yield the output strings. Output in German and English is produced by accessing different dictionaries in a uniform way.

In order to facilitate incremental development of the whole system, our template-based module has a full coverage wrt. the classes of sys-

tem output that are needed. In parallel, we are experimenting with a linguistically more powerful grammar-based generator using OpenCCG[4], an open-source natural language processing environment (Baldridge and Kruijff, 2003). This allows for more fine-grained and controlled choices between linguistic expressions in order to achieve contextually appropriate output.

5.5 Modeling with an Ontology

We use a full model in OWL as the knowledge representation format in the dialogue manager, turn planner and sentence planner. This model includes the entities, properties and relations of the MP3 domain–including the player, data base and playlists. Also, all possible tasks that the user may perform are modeled explicitly. This task model is *user centered* and not simply a model of the application's API.The OWL-based model is transformed automatically to the internal format used in the PATE rule-interpreter.

We use multiple inheritance to model different views of concepts and the corresponding presentation possibilities; e.g., a *song* is a *browsable-object* as well as a *media-object* and thus allows for very different presentations, depending on context. Thereby PATE provides an efficient and elegant way to create more generic presentation planning rules.

6 Experiments and Evaluation

So far we conducted two WOZ *data collection* experiments and one *evaluation* experiment with a baseline version of the SAMMIE system. The SAMMIE-1 WOZ experiment involved only spoken interaction, SAMMIE-2 was multimodal, with speech and haptic input, and the subjects had to perform a primary driving task using a Lane Change simulator (Mattes, 2003) in a half of their experiment session. The wizard was simulating an MP3 player application with access to a large database of information (but not actual music) of more than 150,000 music albums (almost 1 million songs). In order to collect data with a variety of interaction strategies, we used multiple wizards and gave them freedom to decide about their response and its realization. In the multimodal setup in SAMMIE-2, the wizards could also freely decide between mono-modal and multimodal output. (See (Kruijff-Korbayová et al., 2005) for details.)

We have just completed a user evaluation to explore the user-acceptance, usability, and performance of the baseline implementation of the SAMMIE multimodal dialogue system. The users were asked to perform tasks which tested the system functionality. The evaluation analyzed the user's interaction with the baseline system and combined objective measurements like task completion (89%) and subjective ratings from the test subjects (80% positive).

Acknowledgments This work has been carried out in the TALK project, funded by the EU 6th Framework Program, project No. IST-507802.

[4] `http://openccg.sourceforge.net`

References

[Alexandersson and Becker2001] J. Alexandersson and T. Becker. 2001. Overlay as the basic operation for discourse processing in a multimodal dialogue system. In *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, Seattle, Washington, August.

[Baldridge and Kruijff2003] J.M. Baldridge and G.J.M. Kruijff. 2003. Multi-Modal Combinatory Categorial Grammar. In *Proceedings of the 10th Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest, Hungary, April.

[Blaylock and Allen2005] N. Blaylock and J. Allen. 2005. A collaborative problem-solving model of dialogue. In Laila Dybkjær and Wolfgang Minker, editors, *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 200–211, Lisbon, September 2–3.

[Bunt et al.2005] H. Bunt, M. Kipp, M. Maybury, and W. Wahlster. 2005. Fusion and coordination for multimodal interactive information presentation: Roadmap, architecture, tools, semantics. In O. Stock and M. Zancanaro, editors, *Multimodal Intelligent Information Presentation*, volume 27 of *Text, Speech and Language Technology*, pages 325–340. Kluwer Academic.

[Kruijff-Korbayová et al.2005] I. Kruijff-Korbayová, T. Becker, N. Blaylock, C. Gerstenberger, M. Kaißer, P. Poller, J. Schehl, and V. Rieser. 2005. An experiment setup for collecting data for adaptive output planning in a multimodal dialogue system. In *Proc. of ENLG*, pages 191–196.

[Mattes2003] S. Mattes. 2003. The lane-change-task as a tool for driver distraction evaluation. In *Proc. of IGfA*.

[Pfleger et al.2003] N. Pfleger, J. Alexandersson, and T. Becker. 2003. A robust and generic discourse model for multimodal dialogue. In *Proceedings of the 3rd Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, Acapulco.

[Pfleger2004] N. Pfleger. 2004. Context based multimodal fusion. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 265–272, New York, NY, USA. ACM Press.

[Traum and Larsson2003] David R. Traum and Staffan Larsson. 2003. The information state approach to dialog management. In *Current and New Directions in Discourse and Dialog*. Kluwer.

TwicPen : Hand-held Scanner and Translation Software for non-Native Readers

Eric Wehrli
LATL-Dept. of Linguistics
University of Geneva
Eric.Wehrli@lettres.unige.ch

Abstract

TwicPen is a terminology-assistance system for readers of printed (ie. off-line) material in foreign languages. It consists of a hand-held scanner and sophisticated parsing and translation software to provide readers a limited number of translations selected on the basis of a linguistic analysis of the whole scanned text fragment (a phrase, part of the sentence, etc.). The use of a morphological and syntactic parser makes it possible (i) to disambiguate to a large extent the word selected by the user (and hence to drastically reduce the noise in the response), and (ii) to handle expressions (compounds, collocations, idioms), often a major source of difficulty for non-native readers. The system exists for the following language-pairs: English-French, French-English, German-French and Italian-French.

1 Introduction

As a consequence of globalization, a large and increasing number of people must cope with documents in a language other than their own. While readers who do not know the language can find help with machine translation services, people who have a basic fluency in the language while still experiencing some terminological difficulties do not want a full translation but rather more specific help for an unknown term or an opaque expression. Such typical users are the huge crowd of students and scientists around the world who routinely browse documents in English on the Internet or elsewhere. For on-line documents, a variety of terminological tools are available, some of them commercially, such as the ones provided by Google (word translation services) or Babylon Ltd. More advanced, research-oriented systems based on computational linguistics technologies have also been developed, such as GLOSSER-RuG (Nerbonne et al, 1996, 1999), Compass (Breidt et al., 1997) or TWiC (Wehrli, 2003, 2004).

Similar needs are less easy to satisfy when it comes to more traditional documents such as books and other printed material. Multilingual scanning devices have been commercialized[1], but they lack the computational linguistic resources to make them truly useful. The shortcomings of such systems are particularly blatant with inflected languages, or with compound-rich languages such as German, while the inadequate treatment of multiword expressions is obvious for all languages.

TwicPen has been designed to overcome these shortcomings and intends to provide readers of printed material with the same kind and quality of terminological help as is available for on-line documents. For concreteness, we will take our typical user to be a French-speaking reader with knowledge of English and German reading printed material, for instance a novel or a technical document, in English or in German.

For such a user, German vocabulary is likely to be a major source of difficulty due in part to its opacity (for non-Germanic language speakers), the richness of its inflection and, above all, the number and the complexity of its compounds, as exemplified in figure 1 below.[2]

[1] The three main text scanner manufacturers are Whizcom Technologies (http://www.whizcomtech.com), C-Pen (http://www.cpen.com) and Iris Pen (http://www.irislink.com).

[2] See the discussion on "The Longest German Word" on http://german.about.com/library/blwort_long.htm.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 61–64,
Sydney, July 2006.

This paper will describe the TwicPen system, showing how an in-depth linguistic analysis of the sentence in which a problematic word occurs helps to provide a relevant answer to the reader. We will show, in particular, that the advantage of such an approach over a more traditional bilingual terminology system is (i) to reduce the noise with a better selection (disambiguation) of the source word, (ii) to provide in-depth morphological analysis and (iii) to handle multi-word expressions (compounds, collocations, idioms), even when the terms of the expression are not adjacent.

2 Overview of TwicPen

The TwicPen system is a natural follow-up of TWiC (Translation of Words in Context), (see Wehrli, 2003, 2004), which is a system for online terminological help based on a full linguistic analysis of the source material. TwicPen uses a very similar technology, but is available on personal computers (or even PDAs) and uses a hand-held scanner to get the input material. In other words, TwicPen consists of (i) a simple hand-held scanner and (ii) parsing and translation software. TwicPen functions as follows :

- The user scans a fragment of text, which can be as short as one word or as long as a whole sentence or even a whole paragraph.
- The text appears in the user interface of the TwicPen system and is immediately parsed and tagged by the Fips parser described in the next section.
- The user can either position the cursor on the specific word for which help is requested, or navigate word by word in the sentence.
- For each word, the system retrieves from the tagged information the relevant lexeme and consults a bilingual dictionary to get one or several translations, which are then displayed in the user interface.

Figure 1 shows the user interface. The input text is the well-known German compound discussed by Kay et al. (1994) reproduced in (1):

(1) Lebensversicherungsgesellschaftsangestellter
Leben(s)-versicherung(s)-gesellschaft(s)-angestellter
life-insurance-company-employee

Such examples are not at all uncommon in German, in particular in administrative or technical documents.

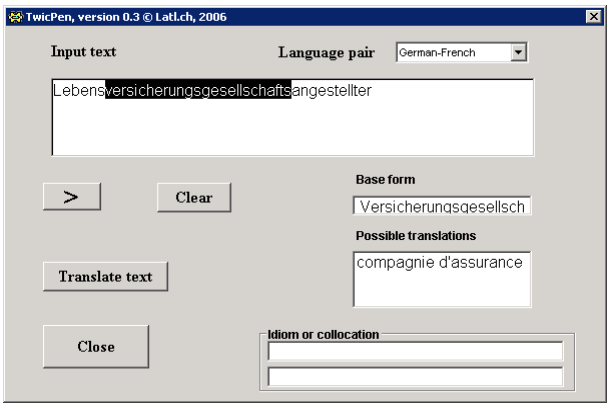


Figure 1: TwicPen user interface with a German compound

Notice that the word *Versicherungsgesellschaft* (English *insurance company* and French *compagnie d'assurance*), which is a compound, has not been analyzed. This is due to the fact that, like many common compounds, it has been lexicalized.

3 The Fips parser

Fips is a robust multilingual parser which is based on generative grammar concepts for its linguistic component and object-oriented design for its implementation. It uses a bottom-up parsing algorithm with parallel treatment of alternatives, as well as heuristics to rank alternatives (and cut their numbers when necessary).

The syntactic structures built by Fips are all of the same pattern, that is : $[_{XP}\ L\ X\ R\ ]$, where L stands for the possibly empty list of left constituents, X for the (possibly empty) head of the phrase and R for the (possibly empty) list of right constituents. The possible values for X are the usual part of speech **Adv**erb, **A**djective, **N**oun, **D**eterminer, **V**erb, **T**ense, **P**reposition, **C**omplementizer, **Inter**jection.

The parser makes use of 3 fundamental mechanisms : projection, merge and move.

3.1 Projection

The projection mechanism assigns a fully developed structure to each incoming word, based on their category and other inherent properties. Thus, a common noun is directly projected to an NP

structure, with the noun as its head, an adjective to an AP structure, a preposition to a PP structure, and so on. We assume that pronouns and, in some languages proper nouns, project to a DP structure (as illustrated in (2a). Furthermore, the occurrence of a tensed verb triggers a more elaborate projection, since a whole TP-VP structure will be assigned. For instance, in French, tensed verbs occur in T position, as illustrated in (2b):

(2)a. [$_{DP}$ Paul], [$_{DP}$ elle]

b. [$_{TP}$ manges$_i$ [$_{VP}$ e$_i$]]

3.2 Merge

The merge mechanism combines two adjacent constituents, A and B, either by attaching constituent A as a left constituent of B, or by attaching B as a right constituent of any active node of A (an active node is one that can still accept subconstituents).

Merge operations are constrained by various, mostly language-specific, conditions which can be described by means of procedural rules. Those rules are stated in a pseudo formalism which attempts to be both intuitive for linguists and relatively straightforward to code (for the time being, this is done manually). The conditions take the form of boolean functions, as described in (3) for left attachments and in (4) for right attachments, where **a** and **b** refer, respectively, to the first and to the second constituent of a merge operation.

(3) **D + T**
a.AgreeWith(b,{number,person})
a.IsArgumentOf(b,subject)

Rule 3 states that a DP constituent (ie. a traditional noun phrase) can (left-)merge with a TP constituent (ie. an inflected verb phrase constituent) if (i) both constituents agree in number and person and (ii) the DP constituent can be interpreted as the subject of the TP constituent.

(4)a. **D + N**
a.HasSelectionFeature(Ncomplement)
b.HasFeature(commonNoun)
a.AgreeWith(b,{number,gender})

b. **V + D**
a.HasFeature(mainVerb)
b.IsArgumentOf(a, directObject)

Rule (4a) states that a common noun can be (right-)attached to a determiner phrase, under the conditions (i) that the head of the DP bears the selectional feature `[+Ncomplement]` (ie. the determiner selects a noun), and (ii) the determiner and the noun agree in gender and number. Finally, rule (4b) allows the attachment of a DP as a right subconstituent of a verb (i) if the verb is not an auxiliary or modal (ie. it is a main verb) and (ii) if the DP can be interpreted as a direct object argument of the verb.

3.3 Move

Although the general architecture of surface structures results from the combination of projection and merge operations, an additional mechanism is necessary to handle so-called extraposed elements and link them to empty constituents (noted **e** in the structural representation below) in canonical positions, thereby creating a chain between the base (canonical) position and the surface (extraposed) position of the "moved" constituent as illustrated in the following example:

(5)a. who did you invite ?

b. [$_{CP}$ [$_{DP}$ who]$_i$ did$_j$ [$_{TP}$ [$_{DP}$ you] e$_j$ [$_{VP}$ invite [$_{DP}$ e]$_i$]]]

4 Multi-word expressions

Perhaps the most advanced feature of TwicPen is its ability to handle multiword expressions (idioms, collocations), including those in which the elements of the expression are not immediately adjacent to each other. Consider the French verb-object collocation *battre-record* (*break-record*), illustrated in (6a, b), as well as in the figure 3.

(6)a. Paul a battu le record national.
Paul broke the national record

b. L'ancien record de Bob Hayes a finalement été battu.
Bob Hayes' old record was finally broken.

The collocation is relatively easy to identify in (6a), where the verb and the direct object noun are almost adjacent and occur in the expected order. It is of course much harder to spot in the (6b) sentence, where the order is reversed (due to passivization) and the distance between the two elements of the collocation is seven words. Nevertheless, as Figure 3 shows, TwicPen is capable of identifying the collocation.

The screenshot given in Figure 3 shows that the user selected the word *battu*, which is a form of

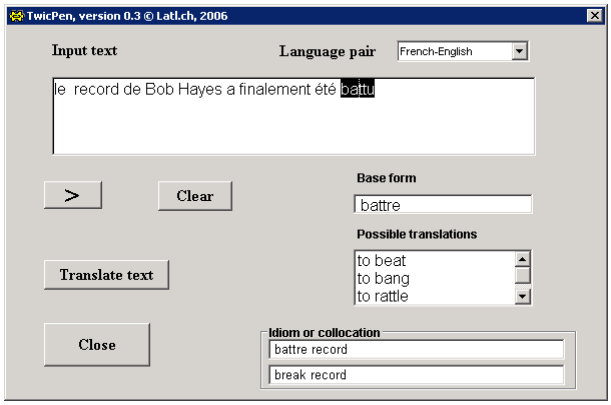


Figure 2: Example of a collocation

the transitive verb *battre*, as indicated in the base form field of the user interface. This lexeme is commonly translated into English as *to beat, to bang, to rattle, etc.*. However, the collocation field shows that *battu* in that sentence is part of the collocation *battre-record* which is translated as *break-record*.

The ability of TwicPen to handle expressions comes from the quality of the linguistic analysis provided by the multilingual Fips parser and of the collocation knowledge base (Seretan et al., 2004). A sample analysis is given in (7b), showing how extraposed elements are connected with canonical empty positions, as assumed by generative linguists.

(7)a. The record that John broke was old.

b. [$_{TP}$ [$_{DP}$ the [$_{NP}$ record$_i$ [$_{CP}$ that$_i$ [$_{TP}$ [$_{DP}$ John] [$_{VP}$ broke [$_{DP}$ e]$_i$]]]]] [$_{VP}$ was [$_{AP}$ old]]]

In this analysis, notice that the noun *record* is coindexed with the relative pronoun *that*, which in turn is coindexed with the empty direct object of the verb *broke*. Given this antecedent-trace chain, it is relatively easy for the system to identify the verb-object collocation *break-record*.

5 Conclusion

Demand for terminological tools for readers of material in a foreign language, either on-line or off-line, is likely to increase with the development of global, multilingual societies. The TwicPen system presented in this paper has been developed for readers of printed material. They scan the sentence (or a fragment of it) containing a word that they don't understand and the system will display (on their laptop) a short list of translations. We have argued that the use of a linguistic parser in such a system brings several major benefits for the word translation task, such as (i) determining the citation form of the word, (ii) drastically reducing word ambiguities, and (iii) identifying multiwords expressions even when their constituents are not adjacent to each other.

Acknowledgement

Thanks to Luka Nerima and Antonio Leoni de Len for their suggestions and comments. The research described in this paper has been supported in part by a grant for the Swiss National Science Foundation (No 101412-103999).

6 References

Breidt, E. and H. Feldweg, 1997. "Accessing Foreign Languages with COMPASS", *Machine Translation*, 12:1-2, 153-174.

Kay, M., M. Gawron and P. Norvig, 1994. *Verbmobil : A Translation System for Face-to-Face Dialog*, Lecture Notes 33, Stanford, CSLI.

Nerbonne, J. and P. Smit, 1996. "GLOSSER-RuG: in Support of Reading in *Proceedings of COLING-1996*, 830-835.

Nerbonne, J. and D. Dokter, 1999. "An Intelligent Word-Based Language Learning Assistant in *TAL* 40:1, 125-142.

Seretan V., Nerima L. and E. Wehrli, 2004. "Multi-word collocation extraction by syntactic composition of collocation bigrams", in Nicolas Nicolov et al. (eds), Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003, Amsterdam, John Benjamins, 91-100.

Wehrli, E. 2003. "Translation of Words in Context", *Proceedings of MT-Summit IX*, New Orleans, 502-504.

Wehrli, E. 2004. "Traduction, traduction de mots, traduction de phrases", in B. Bel et I. Marlien (eds.), *Proceedings of TALN XI*, Fes, 483-491.

An Implemented Description of Japanese: The Lexeed Dictionary and the Hinoki Treebank

Sanae Fujita, Takaaki Tanaka, Francis Bond, Hiromi Nakaiwa
NTT Communication Science Laboratories,
Nippon Telegraph and Telephone Corporation
{sanae, takaaki, bond, nakaiwa}@cslab.kecl.ntt.co.jp

Abstract

In this paper we describe the current state of a new Japanese lexical resource: the Hinoki treebank. The treebank is built from dictionary definition sentences, and uses an HPSG based Japanese grammar to encode both syntactic and semantic information. It is combined with an ontology based on the definition sentences to give a detailed sense level description of the most familiar 28,000 words of Japanese.

1 Introduction

In this paper we describe the current state of a new lexical resource: the Hinoki treebank. The ultimate goal of our research is natural language understanding — we aim to create a system that can parse text into some useful semantic representation. This is an ambitious goal, and this presentation does not present a complete solution, but rather a road-map to the solution, with some progress along the way.

The first phase of the project, which we present here, is to construct a syntactically and semantically annotated corpus based on the machine readable dictionary Lexeed (Kasahara et al., 2004). This is a hand built self-contained lexicon: it consists of headwords and their definitions for the most familiar 28,000 words of Japanese. Each definition and example sentence has been parsed, and the most appropriate analysis selected. Each content word in the sentences has been marked with the appropriate Lexeed sense. The syntactic model is embodied in a grammar, while the semantic model is linked by an ontology. This makes it possible to test the use of similarity and/or semantic class based back-offs for parsing and generation with both symbolic grammars and statistical models.

In order to make the system self sustaining we base the first growth of our treebank on the dictionary definition sentences themselves. We then train a statistical model on the treebank and parse the entire lexicon. From this we induce a thesaurus. We are currently tagging other genres with the same information. We will then use this information and the thesaurus to build a parsing model that combines syntactic and semantic information. We will also produce a richer ontology — for example extracting selectional preferences. In the last phase, we will look at ways of extending our lexicon and ontology to less familiar words.

2 The Lexeed Semantic Database of Japanese

The Lexeed Semantic Database of Japanese consists of all Japanese words with a familiarity greater than or equal to five on a seven point scale (Kasahara et al., 2004). This gives 28,000 words in all, with 46,000 different senses. Definition sentences for these sentences were rewritten to use only the 28,000 familiar words (and some function words). The defining vocabulary is actually 16,900 different words (60% of all possible words). A simplified example entry for the last two senses of the word ドライバー *doraibā* "driver" is given in Figure 1, with English glosses added, but omitting the example sentences. Lexeed itself consists of just the definitions, familiarity and part of speech, all the underlined features are those added by the Hinoki project.

3 The Hinoki Treebank

The structure of our treebank is inspired by the Redwoods treebank of English (Oepen et al., 2002) in which utterances are parsed and the annotator selects the best parse from the full analyses derived by the grammar. We had four main reasons for selecting this approach. The first was that we wanted to develop a precise broad-coverage

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 65–68,
Sydney, July 2006.

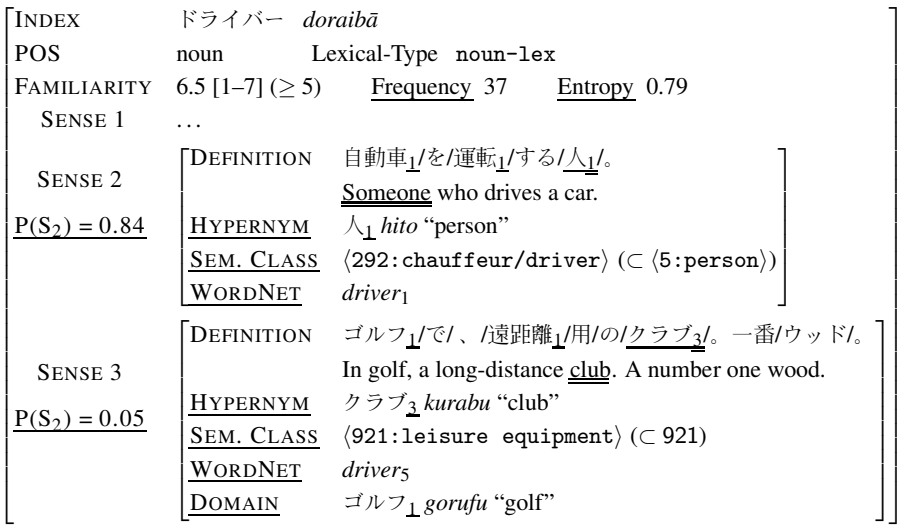


Figure 1: Entry for the Word *doraibā* "driver" (with English glosses)

grammar in tandem with the treebank, as part of our research into natural language understanding. Treebanking the output of the parser allows us to immediately identify problems in the grammar, and improving the grammar directly improves the quality of the treebank in a mutually beneficial feedback loop.

The second reason is that we wanted to annotate to a high level of detail, marking not only dependency and constituent structure but also detailed semantic relations. By using a Japanese grammar (JACY: Siegel (2000)) based on a monostratal theory of grammar (Head Driven Phrase Structure Grammar) we could simultaneously annotate syntactic and semantic structure without overburdening the annotator. The treebank records the complete syntacto-semantic analysis provided by the HPSG grammar, along with an annotator's choice of the most appropriate parse. From this record, all kinds of information can be extracted at various levels of granularity: A simplified example of the labeled tree, minimal recursion semantics representation (MRS) and semantic dependency views for the definition of ドライバー$_2$ *doraibā* "driver" is given in Figure 2.

The third reason was that use of the grammar as a base enforces consistency — all sentences annotated are guaranteed to have well-formed parses. The last reason was the availability of a reasonably robust existing HPSG of Japanese (JACY), and a wide range of open source tools for developing the grammars. We made extensive use of tools from the the Deep Linguistic Processing with HPSG Initiative (DELPH-IN: `http://www.delph-in.net/`) These existing resources enabled us to rapidly develop and test our approach.

3.1 Syntactic Annotation

The construction of the treebank is a two stage process. First, the corpus is parsed (in our case using JACY), and then the annotator selects the correct analysis (or occasionally rejects all analyses). Selection is done through a choice of discriminants. The system selects features that distinguish between different parses, and the annotator selects or rejects the features until only one parse is left. The number of decisions for each sentence is proportional to $\log_2$ in the length of the sentence (Tanaka et al., 2005). Because the disambiguating choices made by the annotators are saved, it is possible to semi-automatically update the treebank when the grammar changes. Re-annotation is only necessary in cases where the parse has become more ambiguous or, more rarely, existing rules or lexical items have changed so much that the system cannot reconstruct the parse.

The Lexeed definition sentences were already POS tagged. We experimented with using the POS tags to mark trees as good or bad (Tanaka et al., 2005). This enabled us to reduce the number of annotator decisions by 20%.

One concern with Redwoods style treebanking is that it is only possible to annotate those trees that the grammar can parse. Sentences for which no analysis had been implemented in the grammar or which fail to parse due to processing constraints are left unannotated. This makes grammar cov-

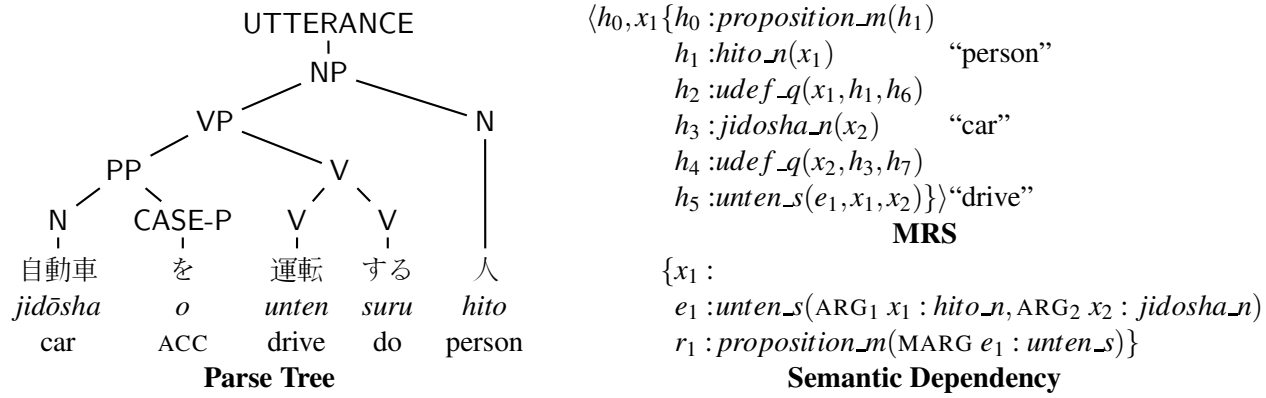


Figure 2: Parse Tree, Simplified MRS and Dependency Views for ドライバー$_2$ *doraibā* "driver"

erage a significant issue. We extended JACY by adding the defining vocabulary, and added some new rules and lexical-types (more detail is given in Bond et al. (2004)). None of the rules are specific to the dictionary domain. The grammatical coverage over all sentences is now 86%. Around 12% of the parsed sentences were rejected by the treebankers due to an incomplete semantic representation. The total size of the treebank is currently 53,600 definition sentences and 36,000 example sentences: 89,600 sentences in total.

3.2 Sense Annotation

All open class words were annotated with their sense by five annotators. Inter-annotator agreement ranges from 0.79 to 0.83. For example, the word クラブ *kurabu* "club" is tagged as sense 3 in the definition sentence for *driver*$_3$, with the meaning "golf-club". For each sense, we calculate the entropy and per sense probabilities over four corpora: the Lexeed definition and example sentences and Newspaper text from the Kyoto University and Senseval 2 corpora (Tanaka et al., 2006).

4 Applications

4.1 Stochastic Parse Ranking

Using the treebanked data, we built a stochastic parse ranking model. The ranker uses a maximum entropy learner to train a PCFG over the parse derivation trees, with the current node, two grandparents and several other conditioning features. A preliminary experiment showed the correct parse is ranked first 69% of the time (10-fold cross validation on 13,000 sentences; evaluated per sentence). We are now experimenting with extensions based on constituent weight, hypernym, semantic class and selectional preferences.

4.2 Ontology Acquisition

To extract hypernyms, we parse the first definition sentence for each sense (Nichols et al., 2005). The parser uses the stochastic parse ranking model learned from the Hinoki treebank, and returns the semantic representation (MRS) of the first ranked parse. In cases where JACY fails to return a parse, we use a dependency parser instead. The highest scoping real predicate is generally the hypernym. For example, for *doraibā*$_2$ the hypernym is 人 *hito* "person" and for *doraibā*$_3$ the hypernym is クラブ *kurabu* "club" (see Figure 1). We also extract other relationships, such as synonym and domain. Because the words are sense tags, we can specialize the relations to relations between senses, rather than just words: ⟨hypernym: doraibā$_3$, *kurabu*$_3$⟩.

Once we have synonym/hypernym relations, we can link the lexicon to other lexical resources. For example, for the manually constructed Japanese ontology **Goi-Taikei** (Ikehara et al., 1997) we link to its semantic classes by the following heuristic: look up the semantic classes C for both the headword (w_i) and hypernym(s) (w_g). If at least one of the index word's classes is subsumed by at least one of the genus' classes, then we consider the relationship confirmed. To link cross-linguistically, we look up the headwords and hypernym(s) in a translation lexicon and compare the set of translations $c_i \subset C(T(w_i))$ with WordNet (Fellbaum, 1998)). Although looking up the translation adds noise, the additional filter of the relationship triple effectively filters it out again.

Adding the ontology to the dictionary interface makes a far more flexible resource. For example, by clicking on the ⟨hypernym: doraibā$_3$, *gorufu*$_1$⟩ link, it is possible to see a list of all the senses re-

lated to golf, a link that is inaccessible in the paper dictionary.

4.3 Semi-Automatic Grammar Documentation

A detailed grammar is a fundamental component for **precise** natural language processing. It provides not only detailed syntactic and morphological information on linguistic expressions but also precise and usually language-independent semantic structures of them. To simplify grammar development, we take a snapshot of the grammar used to treebank in each development cycle. From this we extract information about lexical items and their types from both the grammar and treebank and convert it into an electronically accesible structured database (the lexical-type database: Hashimoto et al., 2005). This allows grammar developers and treebankers to see comprehensive up-to-date information about lexical types, including documentation, syntactic properties (super types, valence, category and so on), usage examples from the treebank and links to other dictionaries.

5 Further Work

We are currently concentrating on three tasks. The first is improving the coverage of the grammar, so that we can parse more sentences to a correct parse. The second is improving the knowledge acquisition, in particular learning other information from the parsed defining sentences — such as lexical-types, semantic association scores, meronyms, and antonyms. The third task is adding the knowledge of hypernyms into the stochastic model.

The Hinoki project is being extended in several ways. For Japanese, we are treebanking other genres, starting with Newspaper text, and increasing the vocabulary, initially by parsing other machine readable dictionaries. We are also extending the approach multilingually with other grammars in the DELPH-IN group. We have started with the English Resource Grammar and the Gnu Contemporary International Dictionary of English and are investigating Korean and Norwegian through cooperation with the Korean Research Grammar and NorSource.

6 Conclusion

In this paper we have described the current state of the Hinoki treebank. We have further showed how it is being used to develop a language-independent system for acquiring thesauruses from machine-readable dictionaries.

With the improved the grammar and ontology, we will use the knowledge learned to extend our model to words not in Lexeed, using definition sentences from machine-readable dictionaries or where they appear within normal text. In this way, we can grow an extensible lexicon and thesaurus from Lexeed.

Acknowledgements

We thank the treebankers, Takayuki Kuribayashi, Tomoko Hirata and Koji Yamashita, for their hard work and attention to detail.

References

Francis Bond, Sanae Fujita, Chikara Hashimoto, Kaname Kasahara, Shigeko Nariyama, Eric Nichols, Akira Ohtani, Takaaki Tanaka, and Shigeaki Amano. 2004. The Hinoki treebank: A treebank for text understanding. In *Proceedings of the First International Joint Conference on Natural Language Processing (IJCNLP-04)*. Springer Verlag. (in press).

Christine Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Chikara Hashimoto, Francis Bond, Takaaki Tanaka, and Melanie Siegel. 2005. Integration of a lexical type database with a linguistically interpreted corpus. In *6th International Workshop on Linguistically Integrated Corpora (LINC-2005)*, pages 31–40. Cheju, Korea.

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1997. *Goi-Taikei — A Japanese Lexicon*. Iwanami Shoten, Tokyo. 5 volumes/CDROM.

Kaname Kasahara, Hiroshi Sato, Francis Bond, Takaaki Tanaka, Sanae Fujita, Tomoko Kanasugi, and Shigeaki Amano. 2004. Construction of a Japanese semantic lexicon: Lexeed. SIG NLC-159, IPSJ, Tokyo. (in Japanese).

Eric Nichols, Francis Bond, and Daniel Flickinger. 2005. Robust ontology acquisition from machine-readable dictionaries. In *Proceedings of the International Joint Conference on Artificial Intelligence IJCAI-2005*, pages 1111–1116. Edinburgh.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christoper D. Manning, Dan Flickinger, and Thorsten Brant. 2002. The LinGO redwoods treebank: Motivation and preliminary applications. In *19th International Conference on Computational Linguistics: COLING-2002*, pages 1253–7. Taipei, Taiwan.

Melanie Siegel. 2000. HPSG analysis of Japanese. In Wolfgang Wahlster, editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 265 – 280. Springer, Berlin, Germany.

Takaaki Tanaka, Francis Bond, and Sanae Fujita. 2006. The Hinoki sensebank — a large-scale word sense tagged corpus of Japanese —. In *Frontiers in Linguistically Annotated Corpora 2006*. Sydney. (ACL Workshop).

Takaaki Tanaka, Francis Bond, Stephan Oepen, and Sanae Fujita. 2005. High precision treebanking – blazing useful trees using POS information. In *ACL-2005*, pages 330–337.

NLTK: The Natural Language Toolkit

Steven Bird
Department of Computer Science and Software Engineering
University of Melbourne, Victoria 3010, AUSTRALIA
Linguistic Data Consortium, University of Pennsylvania,
Philadelphia PA 19104-2653, USA

Abstract

The Natural Language Toolkit is a suite of program modules, data sets and tutorials supporting research and teaching in computational linguistics and natural language processing. NLTK is written in Python and distributed under the GPL open source license. Over the past year the toolkit has been rewritten, simplifying many linguistic data structures and taking advantage of recent enhancements in the Python language. This paper reports on the simplified toolkit and explains how it is used in teaching NLP.

1 Introduction

NLTK, the Natural Language Toolkit, is a suite of Python modules providing many NLP data types, processing tasks, corpus samples and readers, together with animated algorithms, tutorials, and problem sets (Loper and Bird, 2002). Data types include tokens, tags, chunks, trees, and feature structures. Interface definitions and reference implementations are provided for tokenizers, stemmers, taggers (regexp, ngram, Brill), chunkers, parsers (recursive-descent, shift-reduce, chart, probabilistic), clusterers, and classifiers. Corpus samples and readers include: Brown Corpus, CoNLL-2000 Chunking Corpus, CMU Pronunciation Dictionary, NIST IEER Corpus, PP Attachment Corpus, Penn Treebank, and the SIL Shoebox corpus format.

NLTK is ideally suited to students who are learning NLP or conducting research in NLP or closely related areas. NLTK has been used successfully as a teaching tool, as an individual study tool, and as a platform for prototyping and building research systems (Liddy and McCracken, 2005; Sætre et al., 2005).

We chose Python for its shallow learning curve, transparent syntax, and good string-handling. Python permits exploration via its interactive interpreter. As an object-oriented language, Python permits data and code to be encapsulated and re-used easily. Python comes with an extensive library, including tools for graphical programming and numerical processing (Beasley, 2006).

Over the past four years the toolkit grew rapidly and the data structures became significantly more complex. Each new processing task added new requirements on input and output representations. It was not clear how to generalize tasks so they could be applied independently of each other.

As a simple example, consider the independent tasks of tagging and stemming, which both operate on sequences of tokens. If stemming is done first, we lose information required for tagging. If tagging is done first, the stemming must be able to skip over the tags. If both are done independently, we need to be able to align the results. As task combinations multiply, managing the data becomes extremely difficult.

To address this problem, NLTK 1.4 introduced a blackboard architecture for tokens, unifying many data types, and permitting distinct tasks to be run independently. Unfortunately this architecture also came with a significant overhead for programmers, who were often forced to use "rather awkward code structures" (Hearst, 2005). It was clear that the re-engineering done in NLTK 1.4 unduly complicated the programmer's task.

This paper presents a brief overview and tutorial on a new, simplified toolkit, and describes how it is used in teaching.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 69–72,
Sydney, July 2006.

2 Simple Processing Tasks

2.1 Tokenization and Stemming

The following three-line program imports the `tokenize` package, defines a text string, and tokenizes the string on whitespace to create a list of tokens. (NB. '`>>>`' is Python's interactive prompt; '`...`' is the continuation prompt.)

```
>>> text = 'This is a test.'
>>> list(tokenize.whitespace(text))
['This', 'is', 'a', 'test.']
```

Several other tokenizers are provided. We can stem the output of tokenization using the Porter Stemmer as follows:

```
>>> text = 'stemming is exciting'
>>> tokens = tokenize.whitespace(text)
>>> porter = stem.Porter()
>>> for token in tokens:
...     print porter.stem(token),
stem is excit
```

The corpora included with NLTK come with corpus readers that understand the file structure of the corpus, and load the data into Python data structures. For example, the following code reads part *a* of the Brown Corpus. It prints a list of tuples, where each tuple consists of a word and its tag.

```
>>> for sent in brown.tagged('a'):
...     print sent
[('The', 'at'), ('Fulton', 'np-tl'),
('County', 'nn-tl'), ('Grand', 'jj-tl'),
('Jury', 'nn-tl'), ('said', 'vbd'), ...]
```

NLTK provides support for conditional frequency distributions, making it easy to count up items of interest in specified contexts. Such information may be useful for studies in stylistics or in text categorization.

2.2 Tagging

The simplest possible tagger assigns the same tag to each token:

```
>>> my_tagger = tag.Default('nn')
>>> list(my_tagger.tag(tokens))
[('John', 'nn'), ('saw', 'nn'),
 ('3', 'nn'), ('polar', 'nn'),
 ('bears', 'nn'), ('.', 'nn')]
```

On its own, this will tag only 10–20% of the tokens correctly. However, it is a reasonable tagger to use as a default if a more advanced tagger fails to determine a token's tag.

The regular expression tagger assigns a tag to a token according to a series of string patterns. For instance, the following tagger assigns `cd` to cardinal numbers, `nns` to words ending in the letter *s*, and `nn` to everything else:

```
>>> patterns = [
...     (r'\d+(.\d+)?$', 'cd'),
...     (r'\.*s$', 'nns'),
...     (r'.*', 'nn')]
>>> simple_tagger = tag.Regexp(patterns)
>>> list(simple_tagger.tag(tokens))
[('John', 'nn'), ('saw', 'nn'),
 ('3', 'cd'), ('polar', 'nn'),
 ('bears', 'nns'), ('.', 'nn')]
```

The `tag.Unigram` class implements a simple statistical tagging algorithm: for each token, it assigns the tag that is most likely for that token. For example, it will assign the tag `jj` to any occurrence of the word *frequent*, since *frequent* is used as an adjective (e.g. *a frequent word*) more often than it is used as a verb (e.g. *I frequent this cafe*). Before a unigram tagger can be used, it must be trained on a corpus, as shown below for the first section of the Brown Corpus.

```
>>> unigram_tagger = tag.Unigram()
>>> unigram_tagger.train(brown('a'))
```

Once a unigram tagger has been trained, it can be used to tag new text. Note that it assigns the default tag `None` to any token that was not encountered during training.

```
>>> text = "John saw the books on the table"
>>> tokens = list(tokenize.whitespace(text))
>>> list(unigram_tagger.tag(tokens))
[('John', 'np'), ('saw', 'vbd'),
 ('the', 'at'), ('books', None),
 ('on', 'in'), ('the', 'at'),
 ('table', None)]
```

We can instruct the unigram tagger to back off to our default `simple_tagger` when it cannot assign a tag itself. Now all the words are guaranteed to be tagged:

```
>>> unigram_tagger =
...     tag.Unigram(backoff=simple_tagger)
>>> unigram_tagger.train(train_sents)
>>> list(unigram_tagger.tag(tokens))
[('John', 'np'), ('saw', 'vbd'),
 ('the', 'at'), ('books', 'nns'),
 ('on', 'in'), ('the', 'at'),
 ('table', 'nn')]
```

We can go on to define and train a bigram tagger, as shown below:

```
>>> bigram_tagger =\
...     tag.Bigram(backoff=unigram_tagger)
>>> bigram_tagger.train(brown.tagged('a'))
```

We can easily evaluate this tagger against some gold-standard tagged text, using the `tag.accuracy()` function.

NLTK also includes a Brill tagger (contributed by Christopher Maloof) and an HMM tagger (contributed by Trevor Cohn).

3 Chunking and Parsing

Chunking is a technique for shallow syntactic analysis of (tagged) text. Chunk data can be loaded from files that use the common bracket or IOB notations. We can define a regular-expression based chunk parser for use in chunking tagged text. NLTK also supports simple cascading of chunk parsers. Corpus readers for chunked data in Penn Treebank and CoNLL-2000 are provided, along with comprehensive support for evaluation and error analysis.

NLTK provides several parsers for context-free phrase-structure grammars. Grammars can be defined using a series of productions as follows:

```
>>> grammar = cfg.parse_grammar('''
...     S -> NP VP
...     VP -> V NP | V NP PP
...     V -> "saw" | "ate"
...     NP -> "John" | Det N | Det N PP
...     Det -> "a" | "an" | "the" | "my"
...     N -> "dog" | "cat" | "ball"
...     PP -> P NP
...     P -> "on" | "by" | "with"
...     ''')
```

Now we can tokenize and parse a sentence with a recursive descent parser. Note that we avoided left-recursive productions in the above grammar, so that this parser does not get into an infinite loop.

```
>>> text = "John saw a cat with my ball"
>>> sent = list(tokenize.whitespace(text))
>>> rd = parse.RecursiveDescent(grammar)
```

Now we apply it to our sentence, and iterate over all the parses that it generates. Observe that two parses are possible, due to prepositional phrase attachment ambiguity.

```
>>> for p in rd.get_parse_list(sent):
...     print p
(S:
  (NP: 'John')
  (VP:
    (V: 'saw')
    (NP:
      (Det: 'a')
      (N: 'cat')
      (PP: (P: 'with')
        (NP: (Det: 'my') (N: 'ball'))))))
(S:
  (NP: 'John')
  (VP:
    (V: 'saw')
    (NP: (Det: 'a') (N: 'cat'))
    (PP: (P: 'with')
      (NP: (Det: 'my') (N: 'ball')))))
```

The same sentence can be parsed using a grammar with left-recursive productions, so long as we use a chart parser. We can invoke NLTK's chart parser with a bottom-up rule-invocation strategy with `chart.ChartParse(grammar, chart.BU_STRATEGY)`. Tracing can be turned on in order to display each step of the process. NLTK also supports probabilistic context free grammars, and provides a Viterbi-style PCFG parser, together with a suite of bottom-up probabilistic chart parsers.

4 Teaching with NLTK

Natural language processing is often taught within the confines of a single-semester course, either at advanced undergraduate level or at postgraduate level. Unfortunately, it turns out to be rather difficult to cover both the theoretical and practical sides of the subject in such a short span of time. Some courses focus on theory to the exclusion of practical exercises, and deprive students of the challenge and excitement of writing programs to automatically process natural language. Other courses are simply designed to teach programming for linguists, and do not manage to cover any significant NLP content. NLTK was developed to address this problem, making it feasible to cover a substantial amount of theory and practice within a single-semester course.

A significant fraction of any NLP course is made up of fundamental data structures and algorithms. These are usually taught with the help of formal notations and complex diagrams. Large trees and charts are copied onto the board and edited in tedious slow motion, or laboriously prepared for presentation slides. A more effective method is to use live demonstrations in which those diagrams are generated and updated automatically. NLTK provides interactive graphical user interfaces, making it possible to view program state and to study program execution step-by-step (e.g. see Figure 1). Most NLTK components have a demonstration mode, and will perform an interesting task without requiring any special input from the user. It is even possible to make minor modifications to programs in response to "what if" questions. In this way, students learn the mechanics of NLP quickly, gain deeper insights into the data structures and algorithms, and acquire new problem-solving skills. Since these demonstrations are distributed with the toolkit, students can experiment on their own with the algorithms that they have seen presented in class.

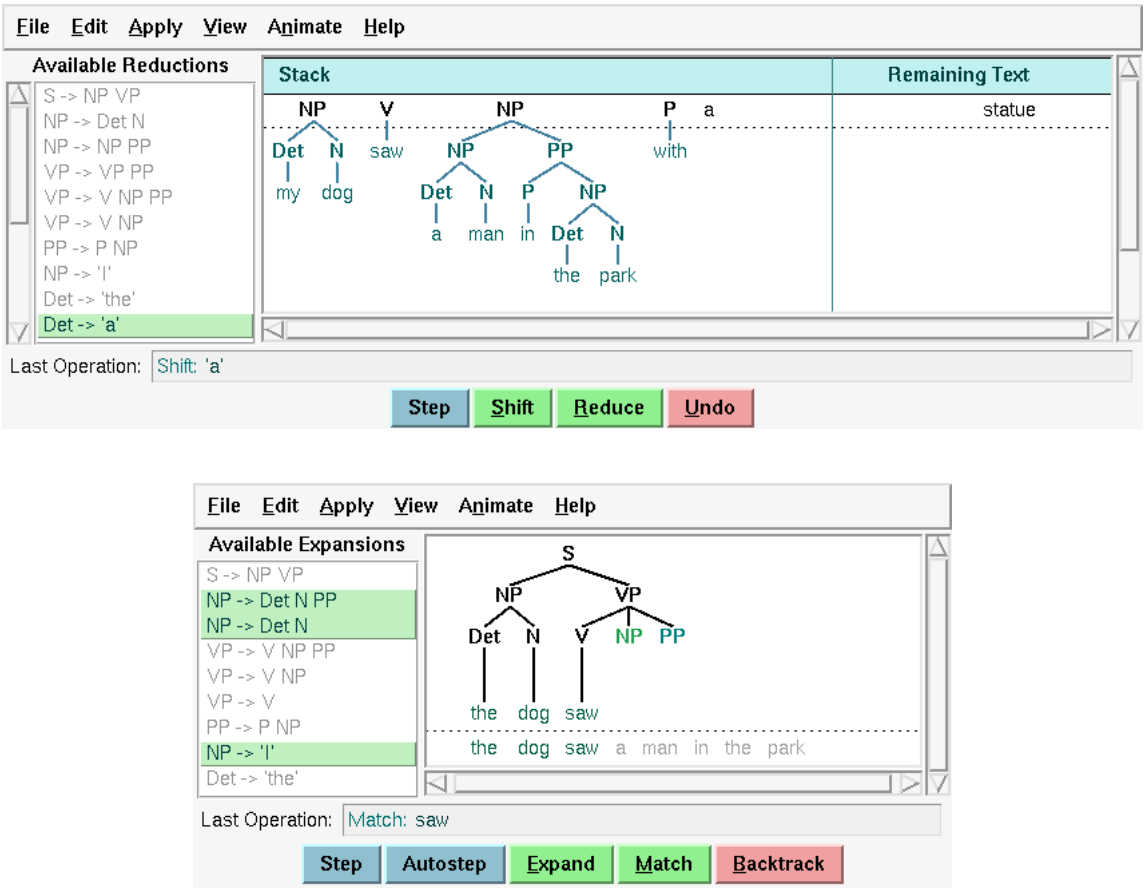


Figure 1: Two Parser Demonstrations: Shift-Reduce and Recursive Descent Parsers

NLTK can be used to create student assignments of varying difficulty and scope. In the simplest assignments, students experiment with one of the existing modules. Once students become more familiar with the toolkit, they can be asked to make minor changes or extensions to an existing module (e.g. build a left-corner parser by modifying the recursive descent parser). A bigger challenge is to develop one or more new modules and integrate them with existing modules to perform a sophisticated NLP task. Here, NLTK provides a useful starting point with its existing components and its extensive tutorials and API documentation.

NLTK is a unique framework for teaching natural language processing. NLTK provides comprehensive support for a first course in NLP which tightly couples theory and practice. Its extensive documentation maximizes the potential for independent learning. For more information, including documentation, download pointers, and links to dozens of courses that have adopted NLTK, please see: `http://nltk.sourceforge.net/`.

Acknowledgements

I am grateful to Edward Loper, co-developer of NLTK, and to dozens of people who have contributed code and provided helpful feedback.

References

Marti Hearst. 2005. Teaching applied natural language processing: Triumphs and tribulations. In *Proc 2nd ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL*, pages 1–8, ACL

Elizabeth Liddy and Nancy McCracken. 2005. Hands-on NLP for an interdisciplinary audience. In *Proc 2nd ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL*, pages 62–68, ACL

Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proc ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 62–69. ACL.

David Beasley. 2006. *Python Essential Reference, 3rd Edition*. Sams.

Rune Sætre, Amund Tveit, Tonje S. Steigedal, and Astrid Lægreid. 2005. Semantic annotation of biomedical literature using Google. In *Data Mining and Bioinformatics Workshop*, volume 3482 of *Lecture Notes in Computer Science*. Springer.

Outilex, a Linguistic Platform for Text Processing

Olivier Blanc
IGM, University of Marne-la-Vallée
5, bd Descartes - Champs/Marne
77454 Marne-la-Vallée, France
oblanc@univ-mlv.fr

Matthieu Constant
IGM, University of Marne-la-Vallée
5, bd Descartes - Champs/Marne
77 454 Marne-la-Vallée, france
mconstan@univ-mlv.fr

Abstract

We present Outilex, a generalist linguistic platform for text processing. The platform includes several modules implementing the main operations for text processing and is designed to use large-coverage Language Resources. These resources (dictionaries, grammars, annotated texts) are formatted into XML, in accordance with current standards. Evaluations on efficiency are given.

1 Credits

This project has been supported by the French Ministry of Industry and the CNRS. Thanks to Sky and Francesca Sigal for their linguistic expertise.

2 Introduction

The Outilex Project (Blanc et al., 2006) aims to develop an open-linguistic platform, including tools, electronic dictionaries and grammars, dedicated to text processing. It is the result of the collaboration of ten French partners, composed of 4 universities and 6 industrial organizations. The project started in 2002 and will end in 2006. The platform which will be made freely available to research, development and industry in April 2007, comprises software components implementing all the fundamental operations of written text processing: text segmentation, morphosyntactic tagging, parsing with grammars and language resource management.

All Language Resources are structured in XML formats, as well as binary formats more adequate to efficient processing; the required format converters are included in the platform. The grammar formalism allows for the combination of statistical approaches with resource-based approaches. Manually constructed lexicons of substantial coverage for French and English, originating from the former LADL[1], will be distributed with the platform under LGPL-LR[2] license.

The platform aims to be a generalist base for diverse processings on text corpora. Furthermore, it uses portable formats and format converters that would allow for combining several software components. There exist a lot of platforms dedicated to NLP, but none are fully satisfactory for various reasons. Intex (Silberztein, 1993), FSM (Mohri et al., 1998) and Xelda[3] are closed source. Unitex (Paumier, 2003), inspired by Intex has its source code under LGPL license[4] but it does not support standard formats for Language Resources (LR). Systems like NLTK (Loper and Bird, 2002) and Gate (Cunningham, 2002) do not offer functionality for Lexical Resource Management.

All the operations described below are implemented in C++ independent modules which interact with each others through XML streams. Each functionality is accessible by programmers through a specified API and by end users through binary programs. Programs can be invoked by a Graphical User Interface implemented in Java. This interface allows the user to define his own processing flow as well as to work on several projects with specific texts, dictionaries and grammars.

[1] French Laboratory for Linguistics and Information Retrieval

[2] Lesser General Public License for Language Resources, http://infolingu.univ-mlv.fr/lgpllr.html.

[3] http://www.dcs.shef.ac.uk/ hamish/dalr/baslow/xelda.pdf.

[4] Lesser General Public License, http://www.gnu.org/copyleft/lesser.html.

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 73–76,
Sydney, July 2006.

3 Text segmentation

The segmentation module takes raw texts or HTML documents as input. It outputs a text segmented into paragraphs, sentences and tokens in an XML format. The HTML tags are kept enclosed in XML elements, which distinguishes them from actual textual data. It is therefore possible to rebuild at any point the original document or a modified version with its original layout. Rules of segmentation in tokens and sentences are based on the categorization of characters defined by the Unicode norm. Each token is associated with information such as its type (word, number, punctuation, ...), its alphabet (Latin, Greek), its case (lowercase word, capitalized word, ...), and other information for the other symbols (opening or closing punctuation symbol, ...). When applied to a corpus of journalistic telegrams of 352,464 tokens, our tokenizer processes 22,185 words per second[5].

4 Morphosyntactic tagging

By using lexicons and grammars, our platform includes the notion of multiword units, and allows for the handling of several types of morphosyntactic ambiguities. Usually, stochastic morphosyntactic taggers (Schmid, 1994; Brill, 1995) do not handle well such notions. However, the use of lexicons by companies working in the domain has much developed over the past few years. That is why Outilex provides a complete set of software components handling operations on lexicons. IGM also contributed to this project by freely distributing a large amount of the LADL lexicons[6] with fine-grained tagsets[7]: for French, 109,912 simple lemmas and 86,337 compound lemmas; for English, 166,150 simple lemmas and 13,361 compound lemmas. These resources are available under LGPL-LR license. Outilex programs are compatible with all European languages using inflection by suffix. Extensions will be necessary for the other types of languages.

Our morphosyntactic tagger takes a segmented text as an input ; each form (simple or compound) is assigned a set of possible tags, extracted from indexed lexicons (cf. section 6). Several lexicons can be applied at the same time. A system of priority allows for the blocking of analyses extracted from lexicons with low priority if the considered form is also present in a lexicon with a higher priority. Therefore, we provide by default a general lexicon proposing a large set of analyses for standard language. The user can, for a specific application, enrich it by means of complementary lexicons and/or filter it with a specialized lexicon for his/her domain. The dictionary look-up can be parameterized to ignore case and diacritics, which can assist the tagger to adapt to the type of processed text (academic papers, web pages, emails, ...). Applied to a corpus of AFP journalistic telegrams with the above mentioned dictionaries, Outilex tags about 6,650 words per second[8].

The result of this operation is an acyclic automaton (sometimes, called word lattice in this context), that represents segmentation and tagging ambiguities. This tagged text can be serialized in an XML format, compatible with the draft model MAF (Morphosyntactic Annotation Framework)(Clément and de la Clergerie, 2005).

All further processing described in the next section will be run on this automaton, possibly modifying it.

5 Text Parsing

Grammatical formalisms are very numerous in NLP. Outilex uses a minimal formalism: Recursive Transition Network (RTN)(Woods, 1970) that are represented in the form of recursive automata (automata that call other automata). The terminal symbols are lexical masks (Blanc and Dister, 2004), which are underspecified word tags i.e. that represent a set of tagged words matching with the specified features (e.g. noun in the plural). Transductions can be put in our RTNs. This can be used, for instance, to insert tags in texts and therefore formalize relations between identified segments.

This formalism allows for the construction of local grammars in the sense of (Gross, 1993). It has been successfully used in different types of applications: information extraction (Poibeau,

[5]This test and further tests have been carried out on a PC with a 2.8 GHz Intel Pentium Processor and a 512 Mb RAM.

[6]http://infolingu.univ-mlv.fr/english/, follow links Linguistic data then Dictionnaries.

[7]For instance, for French, the tagset combines 13 part-of-speech tags, 18 morphological features and several syntactic and semantic features.

[8]4.7 % of the token occurrences were not found in the dictionary; This value falls to 0.4 % if we remove the capitalized occurrences.

The processing time could appear rather slow; but, this task involves not so trivial computations such as conversion between different charsets or approximated look-up using Unicode character properties.

2001; Nakamura, 2005), named entity localization (Krstev et al., 2005), grammatical structure identification (Mason, 2004; Danlos, 2005)). All of these experiments resulted in recall and precision rates equaling the state-of-the-art.

This formalism has been enhanced with weights that are assigned to the automata transitions. Thus, grammars can be integrated into hybrid systems using both statistical methods and methods based on linguistic resources. We call the obtained formalism Weighted Recursive Transition Network (WRTN). These grammars are constructed in the form of graphs with an editor and are saved in an XML format (Sastre, 2005).

Each graph (or automaton) is optimized with epsilon transition removal, determinization and minimization operations. It is also possible to transform a grammar in an equivalent or approximate finite state transducer, by copying the subgraphs into the main automaton. The result generally requires more memory space but can highly accelerate processing.

Our parser is based on Earley algorithm (Earley, 1970) that has been adapted to deal with WRTN (instead of context-free grammar) and a text in the form of an acyclic finite state automaton (instead of a word sequence). The result of the parsing consists of a shared forest of weighted syntactic trees for each sentence. The nodes of the trees are decorated by the possible outputs of the grammar. This shared forest can be processed to get different types of results, such as a list of concordances, an annotated text or a modified text automaton. By applying a noun phrase grammar (Paumier, 2003) on a corpus of AFP journalistic telegrams, our parser processed 12,466 words per second and found 39,468 occurrences.

The platform includes a concordancer that allows for listing in their occurring context different occurrences of the patterns described in the grammar. Concordances can be sorted according to the text order or lexicographic order. The concordancer is a valuable tool for linguists who are interested in finding the different uses of linguistic forms in corpora. It is also of great interest to improve grammars during their construction.

Also included is a module to apply a transducer on a text. It produces a text with the outputs of the grammar inserted in the text or with recognized segments replaced by the outputs. In the case of a weighted grammar, weights are criteria to select between several concurrent analyses. A criterion on the length of the recognized sequences can also be used.

For more complex processes, a variant of this functionality produces an automaton corresponding to the original text automaton with new transitions tagged with the grammar outputs. This process is easily iterable and can then be used for incremental recognition and annotation of longer and longer segments. It can also complete the morphosyntactic tagging for the recognition of semi-frozen lexical units, whose variations are too complex to be enumerated in dictionaries, but can be easily described in local grammars.

Also included is a deep syntactic parser based on unification grammars in the decorated WRTN formalism (Blanc and Constant, 2005). This formalism combines WRTN formalism with functional equations on feature structures. Therefore, complex syntactic phenomena, such as the extraction of a grammatical element or the resolution of some co-references, can be formalized. In addition, the result of the parsing is also a shared forest of syntactic trees. Each tree is associated with a feature structure where are represented grammatical relations between syntactical constituents that have been identified during parsing.

6 Linguistic Resource Management

The reuse of LRs requires flexibility: a lexicon or a grammar is not a static resource. The management of lexicons and grammars implies manual construction and maintenance of resources in a readable format, and compilation of these resources in an operational format. These techniques require strong collaborations between computer scientists and linguists; few systems provide such functionality (Xelda, Intex, Unitex). The Outilex platform provides a complete set of management tools for LRs. For instance, the platform offers an inflection module. This module takes a lexicon of lemmas with syntactic tags as input associated with inflection rules. It produces a lexicon of inflected words associated with morphosyntactic features. In order to accelerate word tagging, these lexicons are then indexed on their inflected forms by using a minimal finite state automaton representation (Revuz, 1991) that allows for both fast look-up procedure and dictionary compression.

7 Conclusion

The Outilex platform in its current version provides all fundamental operations for text processing: processing without lexicon, lexicon and grammar exploitation and LR management. Data are structured both in standard XML formats and in more compact ones. Format converters are included in the platform. The WRTN formalism allows for combining statistical methods with methods based on LRs. The development of the platform required expertise both in computer science and in linguistics. It took into account both needs in fundamental research and applications. In the future, we hope the platform will be extended to other languages and will be enriched with new functionality.

References

Olivier Blanc and Matthieu Constant. 2005. Lexicalization of grammars with parameterized graphs. In *Proc. of RANLP 2005*, pages 117–121, Borovets, Bulgarie, September. INCOMA Ltd.

Olivier Blanc and Anne Dister. 2004. Automates lexicaux avec structure de traits. In *Actes de RECITAL*, pages 23–32.

Olivier Blanc, Matthieu Constant, and Éric Laporte. 2006. Outilex, plate-forme logicielle de traitements de textes écrits. In Cédrick Fairon and Piet Mertens, editors, *Actes de TALN 2006 (Traitement automatique des langues naturelles)*, page to appear, Leuven. ATALA.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Lionel Clément and Éric de la Clergerie. 2005. MAF: a morphosyntactic annotation framework. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 90–94.

Hamish Cunningham. 2002. GATE, a general architecture for text engineering. *Computers and the Humanities*, 36:223–254.

Laurence Danlos. 2005. Automatic recognition of French expletive pronoun occurrences. In *Companion Volume of the International Joint Conference on Natural Language Processing, Jeju, Korea*, page 2013.

Jay Earley. 1970. An efficient context-free parsing algorithm. *Comm. ACM*, 13(2):94–102.

Maurice Gross. 1993. Local grammars and their representation by finite automata. In M. Hoey, editor, *Data, Description, Discourse, Papers on the English Language in honour of John McH Sinclair*, pages 26–38. Harper-Collins, London.

Cvetana Krstev, Duško Vitas, Denis Maurel, and Mickaël Tran. 2005. Multilingual ontology of proper names. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 116–119.

Edward Loper and Steven Bird. 2002. NLTK: the natural language toolkit. In *Proc. of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, Philadelphia*.

Oliver Mason. 2004. Automatic processing of local grammar patterns. In *Proc. of the 7th Annual CLUK (the UK special-interest group for computational linguistics) Research Colloquium*.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 1998. A rational design for a weighted finite-state transducer library. *Lecture Notes in Computer Science*, 1436.

Takuya Nakamura. 2005. Analysing texts in a specific domain with local grammars: The case of stock exchange market reports. In *Linguistic Informatics - State of the Art and the Future*, pages 76–98. Benjamins, Amsterdam/Philadelphia.

Sébastien Paumier. 2003. *De la reconnaissance de formes linguistiques à l'analyse syntaxique. Volume 2, Manuel d'Unitex*. Ph.D. thesis, IGM, Université de Marne-la-Vallée.

Thierry Poibeau. 2001. Extraction d'information dans les bases de données textuelles en génomique au moyen de transducteurs à états finis. In Denis Maurel, editor, *Actes de TALN 2001 (Traitement automatique des langues naturelles)*, pages 295–304, Tours, July. ATALA, Université de Tours.

Dominique Revuz. 1991. *Dictionnaires et lexiques: méthodes et algorithmes*. Ph.D. thesis, Université Paris 7.

Javier M. Sastre. 2005. XML-based representation formats of local grammars for NLP. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 314–317.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. *Proceedings of the International Conference on New Methods in Language Processing*.

Max Silberztein. 1993. *Dictionnaires électroniques et analyse automatique de textes. Le système INTEX*. Masson, Paris. 234 p.

William A. Woods. 1970. Transition network grammars for natural language analysis. *Communications of the ACM*, 13(10):591–606.

The Second Release of the RASP System

Ted Briscoe[†] **John Carroll**[‡] **Rebecca Watson**[†]

[†]Computer Laboratory, University of Cambridge, Cambridge CB3 OFD, UK
firstname.lastname@cl.cam.ac.uk

[‡]Department of Informatics, University of Sussex, Brighton BN1 9QH, UK
J.A.Carroll@sussex.ac.uk

Abstract

We describe the new release of the RASP (robust accurate statistical parsing) system, designed for syntactic annotation of free text. The new version includes a revised and more semantically-motivated output representation, an enhanced grammar and part-of-speech tagger lexicon, and a more flexible and semi-supervised training method for the structural parse ranking model. We evaluate the released version on the WSJ using a relational evaluation scheme, and describe how the new release allows users to enhance performance using (in-domain) lexical information.

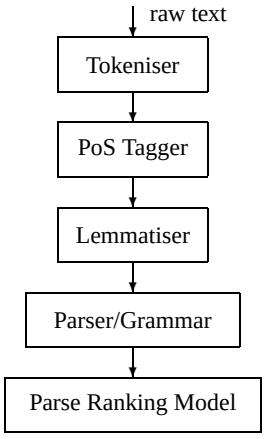


Figure 1: RASP Pipeline

1 Introduction

The first public release of the RASP system (Briscoe & Carroll, 2002) has been downloaded by over 120 sites and used in diverse natural language processing tasks, such as anaphora resolution, word sense disambiguation, identifying rhetorical relations, resolving metonymy, detecting compositionality in phrasal verbs, and diverse applications, such as topic and sentiment classification, text anonymisation, summarisation, information extraction, and open domain question answering. Briscoe & Carroll (2002) give further details about the first release. Briscoe (2006) provides references and more information about extant use of RASP and fully describes the modifications discussed more briefly here.

The new release, which is free for all non-commercial use[1], is designed to address several weaknesses of the extant toolkit. Firstly, all modules have been incrementally improved to cover a greater range of text types. Secondly, the part-of-speech tagger lexicon has been semi-automatically enhanced to better deal with rare or unseen behaviour of known words. Thirdly, better facilities have been provided for user customisation. Fourthly, the grammatical relations output has been redesigned to better support further processing. Finally, the training and tuning of the parse ranking model has been made more flexible.

[1]See http://www.informatics.susx.ac.uk/research/nlp/rasp/ for licence and download details.

2 Components of the System

RASP is implemented as a series of modules written in C and Common Lisp, which are pipelined, working as a series of Unix-style filters. RASP runs on Unix and is compatible with most C compilers and Common Lisp implementations. The public release includes Lisp and C executables for common 32- and 64-bit architectures, shell scripts for running and parameterising the system, documentation, and so forth. An overview of the system is given in Figure 1.

2.1 Sentence Boundary Detection and Tokenisation

The system is designed to take unannotated text or transcribed (and punctuated) speech as input, and not simply to run on pre-tokenised input such as that typically found in corpora produced for NLP purposes. Sentence boundary detection and tokenisation modules, implemented as a set of deterministic finite-state rules in Flex (an open source re-implementation of the original Unix Lex utility)

Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, pages 77–80,
Sydney, July 2006.

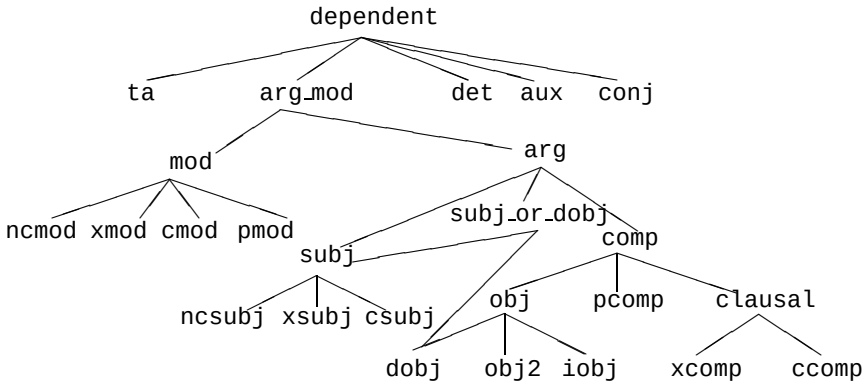


Figure 2: The GR hierarchy

and compiled into C, convert raw ASCII (or Unicode in UTF-8) data into a sequence of sentences in which, for example punctuation tokens are separated from words by spaces, and so forth.

Since the first release this part of the system has been incrementally improved to deal with a greater variety of text types, and handle quotation appropriately. Users are able to modify the rules used and recompile the modules. All RASP modules now accept XML mark up (with certain hard-coded assumptions) so that data can be pre-annotated—for example to identify named entities—before being passed to the tokeniser, allowing for more domain-dependent, potentially multiword tokenisation and classification prior to parsing if desired (e.g. Vlachos *et al.*, 2006), as well as, for example, handling of text with sentence boundaries already determined.

2.2 PoS and Punctuation Tagging

The tokenised text is tagged with one of 150 part-of-speech (PoS) and punctuation labels (derived from the CLAWS tagset). This is done using a first-order ('bigram') hidden markov model (HMM) tagger implemented in C (Elworthy, 1994) and trained on the manually-corrected tagged versions of the Susanne, LOB and (subset of) BNC corpora. The tagger has been augmented with an unknown word model which performs well under most circumstances. However, known but rare words often caused problems as tags for all realisations were rarely present. A series of manually developed rules has been semi-automatically applied to the lexicon to ameliorate this problem by adding further tags with low counts to rare words. The new tagger has an accuracy of just over 97% on the DepBank part of section 23 of the Wall Street Journal, suggesting that this modification has resulted in competitive performance on out-of-domain newspaper text. The tagger implements the Forward-Backward algorithm as well as the Viterbi algorithm, so users can opt for tag thresholding rather than forced-choice tagging (giving >99% tag recall on DepBank, at some cost to overall system speed). Recent experiments suggest that this can lead to a small gain in parse accuracy as well as coverage (Watson, 2006).

2.3 Morphological Analysis

The morphological analyser is also implemented in Flex, with about 1400 finite-state rules incorporating a great deal of lexically exceptional data. These rules are compiled into an efficient C program encoding a deterministic finite state transducer. The analyser takes a word form and CLAWS tag and returns a lemma plus any inflectional affixes. The type and token error rate of the current system is less than 0.07% (Minnen, Carroll and Pearce, 2001). The primary system-internal value of morphological analysis is to enable later modules to use lexical information associated with lemmas, and to facilitate further acquisition of such information from lemmas in parses.

2.4 PoS and Punctuation Sequence Parsing

The manually-developed wide-coverage tag sequence grammar utilised in this version of the parser consists of 689 unification-based phrase structure rules (up from 400 in the first release). The preterminals to this grammar are the PoS and punctuation tags[2]. The terminals are featural descriptions of the preterminals, and the non-terminals project information up the tree using an X-bar scheme with 41 attributes with a maximum of 33 atomic values. Many of the original

[2]The relatively high level of detail in the tagset helps the grammar writer to limit overgeneration and overacceptance.

rules have been replaced with multiple more specific variants to increase precision. In addition, coverage has been extended in various ways, notably to cover quotation and word order permutations associated with direct and indirect quotation, as is common in newspaper text. All rules now have a rule-to-rule declarative specification of the grammatical relations they license (see §2.6). Finally, around 20% of the rules have been manually identified as 'marked' in some way; this can be exploited in customisation and in parse ranking. Users can specify that certain rules should not be used and so to some extent tune the parser to different genres without the need for retraining.

The current version of the grammar finds at least one parse rooted in S for about 85% of the Susanne corpus (used for grammar development), and most of the remainder consists of phrasal fragments marked as independent text sentences in passages of dialogue. The coverage of our WSJ test data is 84%. In cases where there is no parse rooted in S, the parser returns a connected sequence of partial parses covering the input. The criteria are partial parse probability and a preference for longer but non-lexical combinations (Kiefer *et al.*, 1999).

2.5 Generalised LR Parser

A non-deterministic LALR(1) table is constructed automatically from a CF 'backbone' compiled from the feature-based grammar. The parser builds a packed parse forest using this table to guide the actions it performs. Probabilities are associated with subanalyses in the forest via those associated with specific actions in cells of the LR table (Inui *et al.*, 1997). The *n*-best (i.e. most probable) parses can be efficiently extracted by unpacking subanalyses, following pointers to contained subanalyses and choosing alternatives in order of probabilistic ranking. This process backtracks occasionally since unifications are required during the unpacking process and they occasionally fail (see Oepen and Carroll, 2000).

The probabilities of actions in the LR table are computed using bootstrapping methods which utilise an unlabelled bracketing of the Susanne Treebank (Watson *et al.*, 2006). This makes the system more easily retrainable after changes in the grammar and opens up the possibility of quicker tuning to in-domain data. In addition, the structural ranking induced by the parser can be reranked using (in-domain) lexical data which provides conditional probability distributions for the SUBCATegorisation attributes of the major lexical categories. Some generic data is supplied for common verbs, but this can be augmented by user supplied, possibly domain specific files.

2.6 Grammatical Relations Output

The resulting set of ranked parses can be displayed, or passed on for further processing, in a variety of formats which retain varying degrees of information from the full derivations. We originally proposed transforming derivation trees into a set of named grammatical relations (GRs), illustrated as a subsumption hierarchy in Figure 2, as a way of facilitating cross-system evaluation. The revised GR scheme captures those aspects of predicate-argument structure that the system is able to recover and is the most stable and grammar independent representation available. Revisions include a treatment of coordination in which the coordinator is the head in subsuming relations to enable appropriate semantic inferences, and addition of a text adjunct (punctuation) relation to the scheme.

Factoring rooted, directed graphs of GRs into a set of bilexical dependencies makes it possible to compute the transderivational support for a particular relation and thus compute a weighting which takes account both of the probability of derivations yielding a specific relation and of the proportion of such derivations in the forest produced by the parser. A weighted set of GRs from the parse forest is now computed efficiently using a variant of the inside-outside algorithm (Watson *et al.*, 2005).

3 Evaluation

The new system has been evaluated using our reannotation of the PARC dependency bank (DepBank; King *et al.*, 2003)—consisting of 560 sentences chosen randomly from section 23 of the Wall Street Journal—with grammatical relations compatible with our system. Briscoe and Carroll (2006) discuss issues raised by this reannotation.

Relations take the following form: **(relation subtype head dependent initial)** where **relation** specifies the type of relationship between the **head** and **dependent**. The remaining **subtype** and **initial** slots encode additional specifications of the relation type for some relations and the initial or underlying logical relation of the grammatical subject in constructions such as passive. We deter-

mine for each sentence the relations in the test set which are correct at each level of the relational hierarchy. A relation is correct if the head and dependent slots are equal and if the other slots are equal (if specified). If a relation is incorrect at a given level in the hierarchy it may still match for a subsuming relation (if the remaining slots all match); for example, if a **ncmod** relation is mislabelled with **xmod**, it will be correct for all relations which subsume both **ncmod** and **xmod**, e.g. **mod**. Similarly, the GR will be considered incorrect for **xmod** and all relations that subsume **xmod** but not **ncmod**. Thus, the evaluation scheme calculates unlabelled dependency accuracy at the **dependency** (most general) level in the hierarchy. The micro-averaged precision, recall and F_1 score are calculated from the counts for all relations in the hierarchy. The macroaveraged scores are the mean of the individual scores for each relation.

On the reannotated DepBank, the system achieves a microaveraged F_1 score of 76.3% across all relations, using our new training method (Watson *et al.*, 2006). Briscoe and Carroll (2006) show that the system has equivalent accuracy to the PARC XLE parser when the morphosyntactic features in the original DepBank gold standard are taken into account. Figure 3 shows a breakdown of the new system's results by individual relation.

Relation	Precision	Recall	F_1	std GRs
dependent	79.76	77.49	78.61	10696
aux	93.33	91.00	92.15	400
conj	72.39	72.27	72.33	595
ta	42.61	51.37	46.58	292
det	87.73	90.48	89.09	1114
arg_mod	79.18	75.47	77.28	8295
mod	74.43	67.78	70.95	3908
ncmod	75.72	69.94	72.72	3550
xmod	53.21	46.63	49.70	178
cmod	45.95	30.36	36.56	168
pmod	30.77	33.33	32.00	12
arg	77.42	76.45	76.94	4387
subj_or_dobj	82.36	74.51	78.24	3127
subj	78.55	66.91	72.27	1363
ncsubj	79.16	67.06	72.61	1354
xsubj	33.33	28.57	30.77	7
csubj	12.50	50.00	20.00	2
comp	75.89	79.53	77.67	3024
obj	79.49	79.42	79.46	2328
dobj	83.63	79.08	81.29	1764
obj2	23.08	30.00	26.09	20
iobj	70.77	76.10	73.34	544
clausal	60.98	74.40	67.02	672
xcomp	76.88	77.69	77.28	381
ccomp	46.44	69.42	55.55	291
pcomp	72.73	66.67	69.57	24
macroaverage	62.12	63.77	62.94	
microaverage	77.66	74.98	76.29	

Figure 3: Accuracy on DepBank

Acknowledgements

Development has been partially funded by the EPSRC RASP project (GR/N36462 and GR/N36493) and greatly facilitated by Anna Korhonen, Diana McCarthy, Judita Preiss and Andreas Vlachos. Much of the system rests on earlier work on the ANLT or associated tools by Bran Boguraev, David Elworthy, Claire Grover, Kevin Humphries, Guido Minnen, and Larry Piano.

References

Briscoe, E.J. (2006) *An Introduction to Tag Sequence Grammars and the RASP System Parser,* University of Cambridge, Computer Laboratory Technical Report 662.

Briscoe, E.J. and J. Carroll (2002) 'Robust accurate statistical annotation of general text', *Proceedings of the 3rd Int. Conf. on Language Resources and Evaluation (LREC'02),* Las Palmas, Gran Canaria, pp. 1499–1504.

Briscoe, E.J. and J. Carroll (2006) 'Evaluating the Accuracy of an Unlexicalized Statistical Parser on the PARC DepBank', *Proceedings of the COLING/ACL Conference,* Sydney, Australia.

Elworthy, D. (1994) 'Does Baum-Welch re-estimation help taggers?', *Proceedings of the 4th ACL Conference on Applied NLP,* Stuttgart, Germany, pp. 53–58.

Inui, K., V. Sornlertlamvanich, H. Tanaka and T. Tokunaga (1997) 'A new formalization of probabilistic GLR parsing', *Proceedings of the 5th International Workshop on Parsing Technologies (IWPT'97),* MIT, pp. 123–134.

Kiefer, B., H-U. Krieger, J. Carroll and R. Malouf (1999) 'A bag of useful techniques for efficient and robust parsing', *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics,* University of Maryland, pp. 473–480.

King, T.H., R. Crouch, S. Riezler, M. Dalrymple and R. Kaplan (2003) 'The PARC700 Dependency Bank', *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03),* Budapest, Hungary.

Minnen, G., J. Carroll and D. Pearce (2001) 'Applied morphological processing of English', *Natural Language Engineering, vol.7.3,* 225–250.

Oepen, S. and J. Carroll (2000) 'Ambiguity packing in constraint-based parsing — practical results', *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics,* Seattle, WA, pp. 162–169.

Watson, R. (2006) 'Part-of-speech tagging models for parsing', *Proceedings of the 9th Annual CLUK Colloquium,* Open University, Milton Keynes, UK.

Watson, R., E.J. Briscoe and J. Carroll (2006) *Semi-supervised Training of a Statistical Parser from Unlabeled Partially-bracketed Data,* forthcoming.

Watson, R., J. Carroll and E.J. Briscoe (2005) 'Efficient extraction of grammatical relations', *Proceedings of the 9th Int. Workshop on Parsing Technologies (IWPT'05),* Vancouver, Canada.

Author Index