Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 107-126

Using a Generative Model for Sentiment Analysis

Yi Hu*, Ruzhan Lu*, Yuquan Chen*, and Jianyong Duan*

Abstract

This paper presents a generative model based on the language modeling approach for sentiment analysis. By characterizing the semantic orientation of documents as "favorable" (positive) or "unfavorable" (negative), this method captures the subtle information needed in text retrieval. In order to conduct this research, a language model based method is proposed to keep the dependent link between a "term" and other ordinary words in the context of a triggered language model: first, a batch of terms in a domain are identified; second, two different language models representing classifying knowledge for every term are built up from subjective sentences; last, a classifying function based on the generation of a test document is defined for the sentiment analysis. When compared with Support Vector Machine, a popular discriminative model, the language modeling approach performs better on a Chinese digital product review corpus by a 3-fold cross-validation. This result motivates one to consider finding more suitable language models for sentiment detection in future research.

Keywords: Sentiment Analysis, Subjective Sentence, Language Modeling, Supervised Learning.

1. Introduction

Traditional wisdom of document categorization lies in mapping a document to given topics that are usually sport, finance, politics, etc. Whereas, in recent years there has been a growing interest in non-topical analysis, in which characterizations are sought by the opinions and feelings depicted in documents, instead of just their themes. This method of analysis is defined to classify a document as favorable (positive) or unfavorable (negative), which is called sentiment classification. Labeling documents by their semantic orientation provides succinct summaries to readers and will have a great impact on the field of intelligent information retrieval.

* Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dongchuan Rd, Shanghai, China. Tel: 86-21-3420 4591
E-mail: huyi@cs.sjtu.edu.cn

[Received August 2, 2006; Revised March 7, 2007; Accepted March 8, 2007]

In this study, the set of documents is rooted in the topic of digital product review, which will be defined in the latter part of this article. Accordingly, the documents can be classified into praising the core product or criticizing it. Obviously, a praising review corresponds to "favorable" and a criticizing one is "unfavorable" (the neutral review is not considered in this study).

Most research for document categorization adopts the "bag of words" representing model that treats words as independent features. On the other hand, utilizing such a representing mechanism may be imprecise for sentiment analysis. Take a simple sentence in Chinese as an example: "柯达 P712 内部处理器作了升级，处理速度应该更快了。(The processor inside Kodak P712 has been upgraded, so its processing speed ought to be faster.)" The term "柯达 (Kodak)" is very helpful for determining its theme of "digital product review", but words "升级(update)" and "快(fast)" corresponding to "处理器(processor)" and "处理速度(processing speed)" ought to be the important clues for semantic orientation (praise the product). Inversely, see another sentence in Chinese: "这样电池损耗就很快。(So, the battery was used up quickly.)" The words "损耗 (use up)" and "快 (fast)" become unfavorable features of the term "电池 (battery)". That is to say, these words probably contribute less to the sentiment classification if they are dispersed into the document vector, because the direct/indirect relationships between ordinary words and the terms within the sentence are lost. Unfortunately, traditional n-gram features cannot easily deal with these long-distance dependencies.

Sentiment classification is a complex semantic problem [Pang *et al*. 2002; Turney 2002] that needs knowledge for decision-making. The researchers, here, explore a new idea-based language model for the sentiment classification of sentences rather than full document, in which the terms such as "处理器 (processor)", "处理速度 (processing speed)" are target objects to be evaluated in the context. They are mostly the nouns or noun phrases: "屏幕 (Screen)", "分辨率 (Resolution)", "颜色 (Color)", etc. If the sentiment classifying knowledge on how to comment on these terms can be obtained by the training data in advance, the goal of sentiment analysis can be achieved by matching the terms in the test documents. Thus, the classifying task for the full document is changed to recognizing the semantic orientation of all terms in accordance with their sentence-level contexts. This can also be considered a positive/negative word counting method for sentiment analysis.

In this study, the authors construct two language models for each term to capture the difference of sentiment context for that term. In these language models, sentences are divided into terms and their contexts. Sentences without the defined terms are ignored since they make no contribution to the document level sentiment classification; hence, they are omitted from training and test documents. This idea of grouping a document under subjective and objective portions is similar to Pang's work [Pang and Lee 2004].

This work can be divided into three main parts: first, some terms are extracted from a Chinese digital product review corpus [Chen *et al*. 2005]; second, two language models representing positive and negative classifying knowledge for each term are determined from training a subjective sentence set; third, the two models are applied to the test set and then compared with a popular discriminative classifier, SVM. The experiments demonstrate the better performance of the language modeling approach.

The rest of this paper is structured as follows. Section 2 briefly reviews the related works. Section 3 provides short introductions to SVM and language model. Section 4 describes the model in detail. Section 5 presents the method of estimating model parameters, in which a smoothing technique is utilized. Section 6 shows some experiments to exemplify the availability of the language modeling approach. In section 7, conclusions are given.

2. Related Works

A considerable amount of research has been done about document categorization other than topic-based classification in recent years. For example, Biber [Biber 1988] concentrated on sorting documents in terms of their source or source style with stylistic variation such as author, publisher, and native-language background. Sentiment classification for documents, though, has attracted tremendous attention for its broad applications in various domains such as movie reviews and customer feedback reviews [Gamon 2004; Pang *et al*. 2002; Pang and Lee 2004; Turney and Littman 2003]. Many research projects have used positive or negative term counting methods, which automatically determine the positive or negative orientation of a term [Turney and Littman 2002]. Other projects have focused on machine learning algorithms, such as Bayesian Classifier and SVMs, to classify entire reviews in a manner similar to a pattern recognition task.

Some related works focus on categorizing the semantic orientation of individual words or phrases by employing linguistic heuristics [Hatzivassiloglou and McKeown 1997; Hatzivassiloglou and Wiebe 2000; Turney and Littman 2002]. The word's semantic orientation refers to a real number measure of the positive or negative sentiment expressed by a word or a phrase [Hatzivassiloglou and McKeown 1997]. In previous works, the approach taken by Turney [Turney and Littman 2002] is used to derive such values for selected phrases in the document. The semantic orientation of a phrase is determined based on the phrase's Pointwise Mutual Information (PMI) with the words "excellent" and "poor". PMI is defined by Church and Hanks [Church and Hanks 1989] as follows:

$$PMI(w_1 \,\&\, w_2) = \log_2\left(\frac{p(w_1 \,\&\, w_2)}{p(w_1)p(w_2)}\right), \tag{1}$$

where $p(w_1\&w_2)$ is the probability that w_1 and w_2 co-occur. The orientation for a phrase is the difference between its PMI with the word "excellent" and the PMI with the word "poor". The final orientation is:

$$SO(phrase) = PMI(phrase,"excellent") - PMI(phrase,"poor"). \quad (2)$$

This yields values above zero for phrases having greater PMI with the word "excellent" and below zero for greater PMI with "poor". An *SO* value of zero denotes a neutral semantic orientation. This approach is simple but effective. Moreover, it is neither restricted to words of a particular part of speech (*e.g*. adjectives), nor restricted to a single word, but can be applied to multiple-word phrases. The semantic orientation of phrases can be used to determine the sentiment of complete sentences and reviews. In Turney's work, 410 reviews were taken and the accuracy of classifying the documents was found when computing the polarity of phrases for different kinds of reviews. Results ranged from 84% for automobile reviews to as low as 66% for movie reviews.

Another method of classifying documents into positive and negative is to use a learning algorithm to classify the documents. Several algorithms were compared in [Pang *et al*. 2002], where it was found that SVMs generally give better results. Unigrams, bigrams, part of speech information, and the position of the terms in the text are used as features, where using only unigrams is found to produce the best results. Pang *et al*. further analyzed the problem to discover how difficult sentiment analysis is. Their findings indicate that, generally, these algorithms are not able to generate accuracy in the sentiment classification problem in comparison with the standard topic-based categorization. As a method to determine the sentiment of a document, Bayesian belief networks are used to represent a Markov Blanket [Bai 2004], which is a directed acyclic graph where each vertex represents a word and the edges are dependencies between the words.

Methods for extracting subjective expressions from collections are presented in [Pang and Lee 2004]. Subjectivity clues include low-frequency words, collocations, and adjectives and verbs identified using distribution similarity. In [Riloff and Wiebe 2003], a bootstrapping process learns linguistically rich extraction patterns for subjective expressions. Classifiers define unlabeled data to automatically create a large training set, which is then given to an extraction pattern learning algorithm. The learned patterns are then used to identify more subjective sentences. A method to distinguish objective statements from subjective statements is also presented in [Pang and Lee 2004]. This method is based on the assumption that objective and subjective sentences are more possibly to appear in groups. First, each sentence is given a score indicating if the sentence is more likely to be subjective or objective using a Naive Bayes classifier trained on a subjectivity data set. The system then adjusts the subjectivity of a sentence based on how close it is to other subjective or objective sentences.

This method obtains amazing results with up to 86% accuracy on the movie review set. A similar experiment is presented in [Yu and Hatzivassiloglou 2003].

Past works on sentiment-based categorization of entire texts also involve using cognitive linguistics [Hearst 1992; Sack 1994] or manually constructing discriminated lexicons [Das and Chen 2001; Tong 2001]. These works enlighten researchers on the research on learning sentiment models for terms in the given domain.

It is worth referring to an interesting study conducted by Koji Eguchi and Victor Lavrenko [Eguchi and Lavrenko 2006]. In their contribution, they do not pay more attention to sentiment classification itself, but propose several sentiment retrieval models in the framework of generative modeling approach for ranking. Their research assumes that the polarity of sentiment interest is specified in the users' need in some manner, where the topic dependence of the sentiment is considered.

3. SVMs and Language Model

3.1 SVMs

Support Vector Machine (SVM) is highly effective on traditional document categorization [Joachims 1998], and its basic idea is to find the hyper-plane that separates two classes of training examples with the largest margin [Burges 1998]. It is expected that the larger the margin, the better the generalization of the classifier.

The hyper-plane is in a higher dimensional space called feature space and is mapped from the original space. The mapping is done through kernel functions that allow one to compute inner products in the feature space. The key idea in mapping to a higher space is that, in a sufficiently high dimension, data from two categories can always be separated by a hyper-plane. In order to implement the sentiment classification task, these two categories are designated positive and negative. Accordingly, if d is the vector of a document, then the discriminant function is given by:

$$f(d) = w \cdot \phi(d) + b\,. \tag{3}$$

Here, w is the weight vector in feature space that is obtained by the SVM from the training examples. The "·" denotes the inner product and b is a constant. The function ϕ is the mapping function. The equation $w \cdot \phi(d) + b = 0$ represents the hyper-plane in the higher space. Its value $f(d)$ for a document d is proportional to the perpendicular distance of the document's augmented feature vector $\phi(d)$ from the separating hyper-plane. The SVM is trained such that $f(d) \geq 1$ for positive (favorable) examples and $f(x) \leq -1$ for negative (unfavorable) examples.

Joachim's SVM^{light} package [Joachims 1999] was used for training and testing. For more details on SVM, the reader is referred to Cristiani and Shawe-Tailor's tutorial [Cristianini and

Shawe-Taylor 2000] and Roberto Basili's paper [Basili 2003].

3.2 Language Models

A statistical language model is a probability distribution over all possible word sequences in a language [Rosenfeld 2000]. Generally, the task of language modeling handles the problem: how likely would the i^{th} word occur in a sequence given the history of the preceding *i-1* words? In most applications of language modeling, such as speech recognition and information retrieval, the probability of a word sequence is decomposed into a product of n-gram probabilities. Let one assume that L denotes a specified sequence of k words,

$$L = w_1 w_2 ... w_k \ . \tag{4}$$

An n-gram language model considers the sequence L to be a Markov process with probability

$$p(L) = \prod_{i=1}^{k} p(w_i \mid w_{i-n+1}^{i-1}) \ . \tag{5}$$

When n is 1, it is a unigram language model which uses only estimates of the probabilities of individual words, and when n is equal to 2, it is the bigram model which is estimated using information about the co-occurrence of pairs of words. On the other hand, the value of *n-1* is also called the order of the Markov process.

To establish the n-gram language model, probability estimates are typically derived from frequencies of n-gram patterns in the training data. It is common that many possible n-gram patterns would not appear in the actual data used for estimation, even if the size of the data is huge. As a consequence, for a rare or unseen n-gram, the likelihood estimates that are directly based on counts may become problematic. This is often referred to as data sparseness. Smoothing is used to address this problem and has been an important part of various language models.

4. A Generative Model for Sentiment Classification

In this section, a language modeling approach to detect semantic orientation of document is proposed. This approach is very simple: one must observe the usage of language in contexts of terms appearing in positive and negative documents. "Favorable" and "unfavorable" language models are likely to be substantially different: they are prone to different language habits. This divergence in the language models is exploited to effectively classify a test document as positive or negative.

4.1 Two Assumptions

Models usually have their own basic assumptions as foundation of reasoning and calculating, which support their further applications. The researchers also propose two assumptions in this study, and, based on them, employ a language modeling approach to deal with the sentiment classification problem. As mentioned above, ordinary words in a sentence might have correlation with the term in the same sentence. Therefore, this method follows the idea of learning positive and negative language models for each term within sentences. After this, the sentiment classification is transferred into calculating the generation probability of all subjective sentences in a test document by these sentiment models. The following two assumptions are presented:

A_1. A subjective sentence contains at least one sentiment term and is assumed to have obvious semantic orientation.

A_2. A subjective sentence is the processing unit for sentiment analysis.

The first assumption (A_1) gives the definition of subjective sentence, and it means a significant sentence for training or testing should contain at least one term. In contrast, a sentence without any term is regarded as an objective sentence because of its "no contribution" to sentiment. It also assumes that a subjective sentence has complete sentiment information to characterize its own orientation.

The second assumption (A_2) allows one to handle the classification problem of sentence-level processing. Therefore, the authors pay more attention to construct models within the given sentence in terms of this assumption. A_2 is an intuitive idea in many cases.

Previous work has rarely integrated sentence-level subjectivity detection with document-level sentiment polarity. Yu and Hatzivassiloglou [Yu and Hatzivassiloglou 2003] provide methods for sentence-level analysis and for determining whether a sentence is subjective or not, but do not consider document polarity classification. The motivation behind the single sentence selection method of Beineke *et al*. [Beineke *et al*. 2004] is to reveal a document's sentiment polarity, but they do not evaluate the polarity-classification accuracy of results.

4.2 Document Representation

Based on these two assumptions, a document d is naturally reorganized into subjective sentences, and the objective sentences are omitted from d. That is to say, the original d is reduced to:

$$d \triangleq \{s \mid \exists t \in s\} . \tag{6}$$

Furthermore, a subjective sentence can be traditionally represented by a Chinese word sequence as follows,

$$w_1\ w_2 \ldots w_{l-1}\ t_{i,l}\ w_{l+1}\ \ldots w_{l+2}\ w_n\ . \tag{7}$$

In this, "$t_{i,l}$" indicates one term t_i appears in the sentence s_i, which is usually denoted as the serial number 'l' in the sequence. Moreover, the subsequence from w_1 to w_{l-1} is the group of ordinary words on the left side of t_i, and the subsequence from w_{l+1} to w_n is the group of ordinary words on the right. In (7), ordinary words in this sentence consist of t_i's context ($\boldsymbol{Cx_i}$). So, a subjective sentence s_i is simplified to:

$$s_i \triangleq < t_i, Cx_i > . \tag{8}$$

The authors now focus on a special form, by which a document is represented. Let d be defined again,

$$d \triangleq \{< t_i, Cx_i >\} . \tag{9}$$

Definition (9) means that there also exists an independent assumption between sentences and every word has certain correlation with the term within a sentence. Each sentence has semantic orientation and makes a contribution to the global polarity.

Note that it is possible for there to exist more than one term in a sentence. However, when investigating one of them, the others are to be treated as ordinary words. Each term can create a <*t, Cx*> structure. That is to say, one sentence may create more than one such structure.

4.3 Sentiment Models of Term

With respect to each term, each plays an important role in sentiment classification because the pivotal point of this work lies in learning and evaluating its context. This kind of classifying knowledge, derived from the contexts of terms in two subject-sentence collections labeled positive or negative in different contexts, would like to use words with polarity, such as "快 (Fast)" and "慢 (Slow)". A formalized depiction of classifying knowledge is shown as the following 3-tuple k_i:

$$k_i \triangleq < t_i, \theta_i^P, \theta_i^N > \ t_i \in T . \tag{10}$$

The character "T" denotes the list of all terms obtained from collections. With respect to t_i, its classifying knowledge is divided into two models: θ_i^P and θ_i^N which represent the positive and negative models, respectively. The model parameters are estimated from the training data. The contribution of w_j to polarity is quantified by a triggered unigram model to express the long distance dependency, which is a language modeling idea explained in next subsection.

4.4 Language Modeling Approach for Sentiment Classification

Language models applied to information retrieval [Pone and Croft 1998; Song and Croft 1999] have proven the effectiveness of this approach in an ad-hoc IR task. However, little work has been done in sentiment classification other than considering statistical language modeling. The most important idea in this study is to treat sentiment analysis of a document as the comparison of different generation probabilities in their subjective sentences. The difference is derived from the sentiment language models, $\{\theta_i^P\}$ and $\{\theta_i^N\}$, of terms.

Up to the present, the unigram model has been widely used in many applications due to its relatively small parameter space and suitability for avoiding data sparseness. The traditional unigram model takes a strict assumption that each word is independent from all others, consequently, the probability of a word sequence transfers into the product of the probabilities of individual words. In the authors' model, a triggered unigram model based on subjective sentence collection is built. Thus, the sentiment classification of a document becomes a generation process.

It is assumed that each subjective sentence has its own contribution. Therefore, the global document orientation is calculated by the differences between the probabilities of generating every subjective sentence in the document based on the sentiment language models. Thus, the logarithm decision function (11) is defined as:

$$\begin{aligned} F(d;\theta^P,\theta^N) &\triangleq \ln\left(\frac{p(d\mid\theta^P)}{p(d\mid\theta^N)}\right) \\ &= \sum_{t_i\in s_i, s_i\in d}\left(\ln p(s_i\mid t_i,\theta_i^P) - \ln p(s_i\mid t_i,\theta_i^N)\right) \end{aligned} . \tag{11}$$

Equation (11) means that, to a subjective sentence in the document, if it is more possibly generated by the positive language model of term "t_i" than by its negative language model, the sentence gives more weight to positive orientation than the negative. If the opposite is true, the sentence is regarded as more negative. The value of these probabilities is then used to classify the documents:

$$F:\begin{cases} >0 & positive \\ <0 & negative \end{cases} . \tag{12}$$

It is obvious that decision value is the semantic orientation of the whole document. Every subjective sentence will also be calculated by the multiplication of each generation probability of an ordinary word in this sentence except the term itself, *i.e.*:

$$\begin{cases} p(s_i \mid t_i, \theta_i^P) = \prod_{w_j \in Cx_i, w_j \neq t_i} p(w_j \mid t_i, \theta_i^P) \\ p(s_i \mid t_i, \theta_i^N) = \prod_{w_j \in Cx_i, w_j \neq t_i} p(w_j \mid t_i, \theta_i^N) \end{cases}. \tag{13}$$

Using the logarithm, one can rewrite (13) in its final form:

$$\begin{cases} \ln p(s_i \mid t_i, \theta_i^P) = \sum_{w_j \in Cx_i, w_j \neq t_i} \ln p(w_j \mid t_i, \theta_i^P) \\ \ln p(s_i \mid t_i, \theta_i^N) = \sum_{w_j \in Cx_i, w_j \neq t_i} \ln p(w_j \mid t_i, \theta_i^N) \end{cases}. \tag{14}$$

Equations (13) and (14) are both composed of two functions corresponding to positive and negative cases, respectively. Finally, when one substitutes Equation (14) into Equation (11), one gets a new sentiment classifying function:

$$F(d; \theta^P, \theta^N) = \sum_{s_i \in d} \sum_{w_j \in Cx_i, w_j \neq t_i} \ln \left(\frac{p(w_j \mid t_i, \theta_i^P)}{p(w_j \mid t_i, \theta_i^N)} \right). \tag{15}$$

5. Parameter Estimation

In equation (15), one has to estimate $p(w_j \mid t_i, \theta_i^P)$, and $p(w_j \mid t_i, \theta_i^N)$.

5.1 MLE for $p(w_j \mid t_i, \theta_i)$

The researchers have two available training collections labeled with "positive" and "negative". The detailed information of this corpus will be described in Section 6.1.

Two methods are used to estimate the unigram probability: <1> the Maximum Likelihood Estimate (MLE); <2> the Dirichlet Prior Smoothing for language models. The two estimating methods are compared in sentiment classification. The language models are trained on the positive collection (C^P) and negative collection (C^N), respectively. The MLE is

$$\begin{cases} p_{mle}(w_j \mid t_i, \theta_i^P) = \dfrac{\#(< w_j, t_i > \mid w_j \in Cx_i)}{\#(< *, t_i > \mid * \in Cx_i)} & s_i \in C^P \\ p_{mle}(w_j \mid t_i, \theta_i^N) = \dfrac{\#(< w_j, t_i > \mid w_j \in Cx_i)}{\#(< *, t_i > \mid * \in Cx_i)} & s_i \in C^N \end{cases}, \tag{16}$$

where $\#(< w_j, t_i > \mid w_j \in Cx_i)$ is the number of times w_j co-occurring with t_i in same subjective sentences in positive/negative document collection C^P/C^N, while $\#(< *, t_i > \mid * \in Cx_i)$ is the total number of any word (*) co-occurring with the term t_i in the same subjective sentences in C^P/C^N.

In the probability perspective, if a word w_j often co-occurs with t_i in sentences in the training corpus with a positive view, it may mean that it contributes more to a positive orientation than negative, and vice-versa.

The training data consists of small document samples. The MLE models are inherently poor representations of the true models for unseen words that will be unreasonably assigned zero probability. Therefore, a smoothing language model is worthy of being tried to approximate their true models.

5.2 Dirichlet Prior Smoothing

Dirichlet Prior smoothing [Zhai and Lafferty 2001; Zhai and Lafferty 2002] is a general smoothing method for the problem of zero probabilities and is suitable for unigram smoothing. It belongs to a type of linearly interpolated method. The purpose of the Dirichlet Prior smoothing is to address the estimation bias due to the fact that a document collection has a relatively small amount of data used to estimate a unigram model. More specifically, it is designed to discount the *MLE* appropriately and assign non-zero probabilities to n-gram, which are not observed in the collection. This is the normal role of language model smoothing.

The sentence generation is now taken into account. The basic models are the unigram models $\{\theta_i\}$ (includes $\{\theta_i^P\}$ and $\{\theta_i^N\}$, respectively), which will result in models with the Dirichlet Prior smoothing. That is,

$$p_{dir}(w \mid t_i, \theta_i) = \begin{cases} p_\gamma(w \mid t_i, \theta_i) & w \in \{Cx_i\} \\ \alpha\, p_{mle}(w \mid C) & otherwise \end{cases}, \tag{17}$$

where $p_\gamma(w \mid t_i, \theta_i)$ indicates the smoothed probability of w seen in the positive/negative subjective sentence collection of t_i. The probability $p_{mle}(w \mid C)$ denotes the whole corpus (C) language model based on *MLE*, and α is a coefficient controlling the probability mass assigned to unseen words, so that all probabilities sum to one. In general, α may depend on all $p_\gamma(w \mid t_i, \theta_i)$. In this study, the authors exploit the following smoothing formalizations:

$$p_\gamma(w \mid t_i, \theta_i) = \begin{cases} \dfrac{\#(< w, t_i > \mid w \in Cx_i, s_i \in C^P) + \mu\, p_{mle}(w \mid C)}{\#(< *, t_i > \mid * \in Cx_i, s_i \in C^P) + \mu} & to\ \theta_i^P \\ \dfrac{\#(< w, t_i > \mid w \in Cx_i, s_i \in C^N) + \mu\, p_{mle}(w \mid C)}{\#(< *, t_i > \mid * \in Cx_i, s_i \in C^N) + \mu} & to\ \theta_i^N \end{cases}, \tag{18}$$

and

$$\alpha = \frac{\mu}{\mu + \mid C \mid}, \tag{19}$$

where μ is a controlling parameter that needs to be set empirically.

In particular, Dirichlet Prior smoothing may play two different roles in the sentence likelihood generation method. One is to improve the accuracy of the estimated document language model, while the other is to accommodate generation of non-informative common words. The following experiment results further suggest that this smoothing measure is useful in the estimation procedure.

6. Experiment Results and Discussions

This study is interested in the subject of "digital product review", and all documents are obtained from digital product review web sites. In terms of evaluating the results of sentiment classification, the researchers employ average accuracy based on 3-fold cross validation over the polarity corpus in the following several experiments.

6.1 Document Set and Evaluating Measure

The datasets select digital product reviews where the author rating is expressed either with thumbs "up" or thumbs "down". For the works described in this study, the dataset only concentrates on discriminating between positive and negative sentiment.

To avoid domination of the corpus by a small number of prolific reviewers, the corpus imposes a limit of fewer than 25 reviews per author per sentiment category, yielding a corpus of 900 negative and 900 positive reviews, with a total of more than a hundred reviewers represented. Some statistics about the corpus are shown in Table 1.

Table 1. The two collections from the same domain (digital product reivew).

Collections	# of Documents	Average # of Subjective Sentences	Sizes (KB)
Positive	900	28.3	462.99
Negative	900	25.9	453.82

Note that these 1800 documents in the corpus have obvious semantic orientations to their products: favorable or unfavorable. Furthermore, in terms of positive documents, they contain an average of 28.3 subjective sentences, while negative document collections contain an average of 25.9. All these digital product reviews downloaded from several web sites are about electronic products, such as DV, mobile phones, and cameras. On the other hand, all of these Chinese documents have been pre-processed in a standard manner: they are segmented into words and Chinese stop words are removed. All of these labeled documents are to be naturally divided into three collections in every process of 3-fold cross validation, which are used either for training or for testing.

In evaluating processes, a document may be grouped into positive or negative. That is to say, there exist two kinds of classification errors called "false negative" and "false positive".

Thus, the authors could build the following Contingency Table.

Table 2. Contingency Table.

	Tagged Positive	Tagged Negative
True Positive	A	B
True Negative	C	D

In the table A, B, C and D respectively indicate the number of every case. When the system classifies a true positive document into "positive" or classifies a true negative document into "negative", these two are correct, yet the other two cases are wrong. Therefore, the accuracy is defined as a global evaluation mechanism:

$$Accuracy = (A + D)/(A + B + C + D)\ . \tag{20}$$

Obviously, the larger the accuracy value is, the better the system performance is. In the following experiments, the 3-fold cross validation based average accuracy is the major evaluating measure in the following experiments.

6.2 Term Extraction

The researchers extract term candidates using a term extractor from the previous work of the authors [Chen *et al*. 2005]. Following this study, the hybrid method for automatic extraction of terms from domain-specific un-annotated Chinese corpus is used through means of linguistic knowledge and statistical techniques. Then, hundreds of terms applied in the sentiment analysis are extracted from the digital product review documents. They are ranked by their topic-relativity scores.

The main idea in [Chen *et al*. 2005] lies in finding the two neighboring Chinese characters with high co-occurrence, called "bi-character seeds". These seeds can only be terms or the components of terms. For instance, the seed "分辨" is the left part of the real term "分辨率 (Resolution)". So the system has to determine the two boundaries by adding characters one by one to these seeds in both directions to acquire multi-character term candidates. Apparently, there exist many non-terms in these candidates, so one must take a dual filtering strategy and introduce a weighting formula to filter these term candidates via a large background corpus.

Although the authors have adopted the dual filtering strategy in this system to improve performance, it cannot separate the terms and non-terms completely. Therefore, it also needs manual selection of the suitable terms that strictly belong to the digital product domain. The terms were chosen from the candidate list one by one via their topic-relativity scores.

It is worth noting that all the selected terms are nouns/noun phrases that represent concepts that are usually evaluated in real-life contexts. For example, "数码相机 (digital

camera, one of the digital products)", "处理器 (processor, a key part of some digital products)".

6.3 Experiments and Discussions

Three experiments were designed to investigate the proposed method as compared to SVM. The first was to select the most suitable number of terms given their topic-relativity to the domain. The second was to select a suitable kernel from linear, polynomial, RBF and sigmoid kernels for sentiment classification. The last was to compare the performance between the language modeling approach and SVM.

With respect to these three experiments, the 1800 digital product reviews were split into three parts: 1000 training samples (500 positive and 500 negative); 600 test samples (300 positive and 300 negative); and the remaining 200 samples (100 positive and 100 negative) that were prepared for choosing a suitable number of terms.

Table 3 shows a series of contrastive results by testing on the 200 samples after training models of terms ranging from 20 to 200 given their topic-relativity ranks. This is a method for selecting a suitable term set. In this experiment, unigram models are employed by MLE. Here, all of the Chinese words occurring are used as unigrams to learn the language models, and this is different from selecting a portion of them in the following experiments (see Section 6.4).

Table 3. Average accuracy based on the number of terms from 20 to 200 according to their topic-relativity ranking scores. In this experiment, we employ the unigram model by MLE.

# of terms	20	40	60	80	100	120	**140**	160	180	200
Avg. Accuracy	48.31	50.50	57.11	58.78	70.83	74.27	**79.31**	77.04	76.78	73.50

The experiment proves that it is not clear whether or not one ought to use a large term set for achieving better system performance, because redundant terms may bring "noise" to semantic polarity decision. As seen in Table 3, experimental results achieve the greatest accuracy when keeping 140 terms by topic-relativity ranking scores in the term set. According to this result, the authors use the 140 terms next for smoothing of sentiment language models and comparison with SVM.

6.4 Comparison with SVM

Unigrams are extracted as input feature sets for SVM. The following experiments compare the performance of SVM using linear, polynomial, RBF and sigmoid kernels, the four conventional learning methods commonly used for text categorization. The SVM^{light} package [Joachims 1999] was used for training and testing on the document-level, and other

parameters of different kernel functions were set to their default values in this package. This experiment aims at exploring which method is more suitable for the sentiment detection problem (See Table 4).

To make sure that the results for the four kernels are not biased by an inappropriate choice of features, all four methods are run after selecting unigrams (Chinese words) appearing at least three times in the whole 1800 document collection. Finally, the total number of features in this study is 5783 for SVM, including those "terms" used in the language modeling approach.

Table 4. Comparison of four kernel functions on the digital product review training and test corpus and average performance over two categories. Linear kernel achieves highest performance on unigram feature set.

Features	# of features	Linear	Polynomial	Radial Basis Function	Sigmoid
unigrams	5783	**80.17**	61.25	53.09	51.26

The result with the best performance in the test set is the linear kernel. Thus, the language model based method is compared with the SVM using linear kernel. The next table gives the results achieved by the language modeling approach and the control group. In this experiment, the 5783 single word forms (*i.e*. vocabulary) are also used as the features for language models.

Table 5. Comparison between language model based method and SVM using linear kernel.

	# of features	AvgAccuracy	% change over SVM
SVM (Linear Kernel)	5783	80.17	—
Uni-MLE	5783	83.10	+3.65
Uni-Smooth (μ=1100)	5783	**85.33**	+6.44

Seen from table 5, Uni-MLE performs better on the unigrams features set than SVM, which achieved an average significant improvement of 3.65% compared with the best SVM result. As to the model smoothing, Dirichlet Prior smoothes unigram language model with parameter μ set to 1100 (In this experiment, the best result appears when $\mu = 1100$ in Dirichlet Prior smoothing). It makes a contribution to estimating a better unigram language model leading to a significantly better result than SVM (+6.44%). The effect of the smoothing method in sentiment analysis is just like its effect on most language model based applications in NLP. In practice, the unigram model built up from the two limited collections by simple MLE has not enough reasonability in terms of the unseen words. The smoothing method gives the unobserved ordinary words of every term a suitable non-zero probability and improves the system performance.

The better results obtained by this generative model may be due to the sentiment

description within sentences, which proves that the two assumptions in Section 4.1 may be reasonable. The authors use the triggered unigram models to describe the classifying contribution of features of every term, and then construct sentiment language models. Accordingly, the motivation to further explore the refinement of sentiment language models based on learning higher order models and introduce more powerful smoothing methods in future is acquired.

7. Conclusions

In this paper, the authors have presented a new language modeling approach for sentiment classification. To this generative model, the terms of a domain are introduced as counting terms, and their contexts are learnt to create sentiment language models. It was assumed that sentences have complete semantic orientation when they contain at least one term. This assumption allows one to design models to learn positive and negative language models from the subjective sentence set with polarity. The approach is then used to test a real document in steps: first to generate all the subjective sentences in the document, and then to generate each ordinary word in turn depending on the terms by positive and negative sentiment models. The difference between the generation probabilities by the two models is used as the determining rule for sentiment classification.

The authors have also discussed how the proposed model resolves the sentiment classification problem by refining the basic unigram model through smoothing. When the language model based method is compared with a popular discriminative model, *i.e*., SVM, the experiment shows the potential power of language modeling. It was demonstrated that the proposed method is applicable for learning the positive and negative contextual knowledge effectively in a supervised manner.

The difficulty of sentiment classification is apparent: negative reviews may contain many apparently positive unigrams even while maintaining a strongly negative tone and vice-versa. In terms of the Chinese language, it is a language of concept combination, allowing the usage of words to be more flexible than in Indo-European languages, which makes it more difficult to acquire statistic information than other languages. All classifiers will face this difficulty. Therefore, the authors plan to improve the language model based method in the following three possibilities:

Future works may focus on finding a good way to estimate better language models, especially the higher order n-gram models and more powerful smoothing methods.

The authors have assumed an independent condition among sentences so far. It is also possible to introduce a suitable mathematic model to group the close sentences. Constructing an enlarged sentiment analyzing area may utilize more linking information between words.

The conceptual analysis of Chinese words may be helpful to sentiment analysis because this theory pays more attention to counting the real sense of concepts. In future works, the authors may integrate more conceptual features into the models.

Acknowledgement

This work is supported by NSFC Major Research Program 60496326: Basic Theory and Core Techniques of Non Canonical Knowledge.

References

Bai, X., R. Padman, and E. Airoldi, "Sentiment extraction from unstructured text using tabu search-enhanced markov blanket," In *Proceedings of the International Workshop on Mining for and from the Semantic Web*, 2004, Seattle, WA, USA.

Basili, R., "Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms by Thorsten Joachims," *Computational Linguistics*, 29(4), 2003, pp. 655-661.

Beineke, P., T. Hastie, C. Manning, and S. Vaithyanathan, "Exploring sentiment summarization," In *AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications* (AAAI tech report SS-04-07), 2004.

Biber, D., *Variation across Speech and Writing*, The Cambridge University Press, 1988.

Burges, C., "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, 2(2), 1998, pp. 121-167.

Chen, X., X. Li, Y. Hu, and R. Lu, "Dual Filtering Strategy for Chinese Term Extraction," In *Proceedings of FSKD(2)*, Changsha, China, 2005, pp. 778-786.

Church, K. W., and P. Hanks, "Word association norms, mutual information and lexicography," In *Proceedings of the 27th Annual Conference of the ACL*, 1989, Vancouver, BC, Canada.

Cristianini, N., and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*, The Cambridge University Press, 2000.

Das, S., and M. Chen, "Yahoo! for Amazon: Extracting market sentiment from stock message boards," In *Proceedings of the 8th Asia Pacific Finance Association Annual Conference*, 2001, Bangkok, Thailand.

Eguchi, K., and V. Lavrenko, "Sentiment Retrieval using Generative Models," In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2006, Sydney, Australia, pp. 345-354.

Gamon, M., "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," In *Proceedings the 20th International Conference on Computational Linguistics*, 2004, Switzerland.

Hatzivassiloglou, V., and K. McKeown, "Predicting the semantic orientation of adjectives," In *Proceedings of the 35th ACL/8th EACL*, 1997, Madrid, Spain, pp. 174-181.

Hatzivassiloglou, V., and J. Wiebe, "Effects of Adjective Orientation and Gradability on Sentence Subjectivity," In *Proceedings the 18th International Conference on Computational Linguistics*, 2000, Germany, pp. 299-305.

Hearst, M., "Direction-based text interpretation as an information access refinement," *Text-based intelligent systems: current research and practice in information extraction and retrieval*, ed. by Paul Jacobs, Lawrence Erlbaum Associates, 1992, pp. 257-274.

Joachims, T., "Text categorization with support vector machines: Learning with many relevant features," In *Proceedings of the European Conference on Machine Learning*, 1998, Chemnitz, pp. 137-142.

Joachims, T., "Making large-scale SVM learning practical", *Advances in Kernel Methods-Support Vector Learning*, ed. by Bernhard Scholkopf and Alexander Smola, The MIT Press, 1999, pp. 44-56.

Pang, B., and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," In *Proceedings of the 42nd ACL*, 2004, Barcelona, Spain, pp. 271-278.

Pang, B., L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment Classification using Machine Learning Techniques," In *Proceedings of The Conference on Empirical Methods in Natural Language Processing,* 2002, Philadelphia, USA.

Pone, J., and W. B. Croft, "A language modeling approach to information retrieval," In *Proceedings of the 21st Annual Int'l ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998, Melbourne, Australia.

Riloff, E., and J. Wiebe, "Learning extraction patterns for subjective expressions," In *Proceedings of the 41st Conference on Empirical Methods in Natural Language Processing*, 2003, Sapporo, Japan, pp. 105-112.

Rosenfeld, R., "Two decades of statistical language modeling: where do we go from here?" In *Proceedings of the IEEE*, 88(8), 2000.

Sack, W., "On the computation of point of view," In *Proceedings of the Twelfth AAAI, Student abstract*, 1994, Seattle, WA, USA, pp. 1488.

Song, F., and W. B. Croft, "A general language model information retrieval," In *Proceedings of the 22nd Annual Int'l ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999, Berkeley, CA, USA.

Tong, R.M., "An operational system for detecting and tracking opinions in on-line discussion," Workshop Notes, *SIGIR Workshop on Operational Text Classification*, 2001, New Orleans.

Turney, P.D., "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews," In *Proceedings of the ACL*, 2002, Philadelphia, Pennsylvania, USA, pp. 417-424.

Turney, P.D., and M. L. Littman, "Measuring praise and criticism: Inference of semantic orientation from association," *ACM Transactions on Information Systems (TOIS)*, 21(4), 2003, pp. 315-346.

Turney, P.D., and M. L. Littman, "Unsupervised learning of semantic orientation from a hundred-billion-word corpus," Technical Report EGB-1094, National Research Council, Canada, 2002.

Yu, H., and V. Hatzivassiloglou, "Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences," In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003, Sapporo, Japan.

Zhai, C. and J. Lafferty, "A study of smoothing methods for language models applied to ad hoc information retrieval," In *Proceedings of SIGIR*, 2001, New Orleans, USA.

Zhai, C. and J. Lafferty, "Two Stage Language Models for Information Retrieval," In *Proceedings of S IGIR*, 2002, Tampere, Finland.

Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 127-154

An Empirical Study of Non-Stationary Ngram Model and its Smoothing Techniques

Jinghui Xiao*, Bingquan Liu* and Xiaolong Wang*

Abstract

Recently many new techniques have been proposed for language modeling, such as ME, MEMM and CRF. However, the ngram model is still a staple in practical applications. It is well worthy of studying how to improve the performance of the ngram model. This paper enhances the traditional ngram model by relaxing the stationary hypothesis on the Markov chain and exploiting the word positional information. Such an assumption is made that the probability of the current word is determined not only by history words but also by the words positions in the sentence. The non-stationary ngram model (NS ngram model) is proposed. Several related issues are discussed in detail, including the definition of the NS ngram model, the representation of the word positional information and the estimation of the conditional probability. In addition, three smoothing approaches are proposed to solve the data sparseness problem of the NS ngram model. Several smoothing algorithms are presented in each approach. In the experiments, the NS ngram model is evaluated on the pinyin-to-character conversion task which is the core technique of the Chinese text input method. Experimental results show that the NS ngram model outperforms the traditional ngram model significantly by the exploitation of the word positional information. In addition, the proposed smoothing techniques solve the data sparseness problem of the NS ngram model effectively with great error rate reduction.

Keywords: Ngram, Stationary Hypothesis, Pinyin-to-character Conversion, Smoothing

1. Introduction

Statistical language model plays an important role in natural language processing. It has a wide range of applications in many domains, such as speech recognition [Jelinek 1997], OCR [Kolak *et al*. 2003], machine translation [Brown *et al*. 1992], and pinyin-to-character

* School of Computer Science and Techniques, Harbin Institute of Technology, Harbin, 150001, China
E-mail: {xiaojinghui, liubq, wangxl}@insun.hit.edu.cn

[Received November 28, 2006; Revised April 19, 2007; Accepted April 23, 2007]

conversion [Gao *et al*. 2005; Xiao *et al*. 2005] etc. In recent years, great efforts are devoted to the research of language modeling. Many novel techniques are proposed, such as maximum entropy model [Rosenfeld 1994], maximum entropy Markov model [McCallum *et al*. 2000] and conditional random field model [Lafferty *et al*. 2001]. However, the ngram model is still a staple in practical applications. Therefore, it is well worthy of studying how to improve the performance of the ngram model.

The ngram model takes the word sequence as a Markov chain. It makes the Markov hypothesis on the sequence so as to simplify the probability inference. There are actually two hypotheses implied by the Markov hypothesis, named the limited history hypothesis and the stationary hypothesis [Manning and Schutze 1999]. The first one assumes that the probability of the current word is determined only by a few of previous words, but irrelevant to the whole history of words. The second one assumes that the word probability is irrelevant to the actual word positions in the sentence.

The most obvious extension to the traditional ngram model is simply to enlarge the number of history words and build up the higher-order ngram model [Carpenter 2005]. However, the high-order ngram model suffers from the curse of dimensionality [Novak and Ritter 1998]. The bigram model and the trigram model are currently two prevalent language models.

From another point of view, the paper relaxes the stationary hypothesis and enhances the traditional ngram model by exploiting the word positional information. It is based on the philosophy that most words are not only constrained by their contextual information, but also influenced by their positions in the sentence. For example, the Chinese word "首先" (first of all) is usually used to start a sentence, but rarely occurs elsewhere in the sentence. Then higher probability should be assigned to it by a language model when it is in the front of a sentence, and lower probability elsewhere. Moreover, some of punctuations, such as full stop and exclamation, always appear at the end of a sentence. So it may be mistaken for a Chinese sentence that the exclamation appears in the middle of it. Therefore, a language model can benefit from modeling the word positional information.

This paper enhances the traditional ngram model by the exploitation of the word positional information. The non-stationary ngram model (NS ngram model) is proposed. Several related issues are discussed in detail, including the definition of the NS ngram model, the representation of the word positional information and the estimation of the conditional probability. In addition, three smoothing approaches are proposed to solve the data sparseness problem of the NS ngram model. The NS ngram model is evaluated on the pinyin-to-character conversion task which is the core technique of the Chinese text input method. Experimental results show that the NS ngram model outperforms the traditional ngram model significantly and the smoothing techniques proposed in this paper solve the data sparseness problem of the

NS ngram model effectively with great error rate reduction.

The remaining part of the paper is organized as follows. The related works are outlined in section 2. In section 3, the NS ngram model is proposed and several related issues are discussed in detail. In section 4, the data sparseness problem of the NS ngram problem is addressed and three smoothing approaches are proposed. The experimental results and discussions are presented in section 5 and the conclusion is drawn in section 6.

2. Related Works

There are many ways to improve the performance of the ngram model. The most obvious way is to relax the limited history hypothesis and build up the high-order ngram model, which has been discussed in the above section. Another way is to construct the skipping ngram model [Rosenfeld 1994; Ney *et al*. 1994], in which the current word is constrained by the skipped words in the word history, other than the adjacent words. The skipping ngram model can exploit more information of history words and avoid the curse of dimensionality meanwhile. In the experiments, it yields limited improvements by interpolating with the traditional ngram model.

The class-based ngram model [Brown *et al*. 1992] is constructed based on word cluster instead of word. The syntax and semantic information can be well captured in this way. Meanwhile, the parameter space is reduced greatly and the data sparseness problem is alleviated. However, the predictive capability of the class-based ngram model is much lower than the traditional ngram model due to its small parameter space. It usually achieves limited improvements by interpolating with the traditional ngram model.

The cache-based ngram model [Kuhn 1988; Kuhn and Mori 1990] assumes that people tends to use words as few as possible in the article. If a word has been used, it would possibly be used again in the future. The cache-based ngram model is usually utilized to construct a self-adaptive language model.

3. Non-Stationary Ngram Model

This section firstly reviews the traditional ngram model briefly. Secondly, it defines the NS ngram model formally. Thirdly, the word positional information is formalized. Finally, the estimation method is provided for the conditional probability of the NS ngram model.

3.1 Ngram Model

Language model aims to determine the probability of the sequence of words. The sequence probability is usually decomposed into the conditional probabilities of words which are composed of sequences. For the sequence of $s = w_{l_1,p_1} w_{l_2,p_2} ... w_{l_m,p_m}$, its probability is

calculated in formula (1):

$$P(s) = \prod_{i=1}^{m} p(w_{l_i,p_i} \mid w_{l_1,p_1}, w_{l_2,p_2} ... w_{l_{i-1},p_{i-1}}), \quad (1)$$

where w_{l_i,p_j} is the i^{th} word in the lexicon and appears at the j^{th} position in sequence S.

The ngram model makes the Markov hypothesis on the sequence so as to simplify formula (1). The procedures are described in formula (2):

$$P(s) \approx \prod_{i=1}^{m} p(w_{l_i,p_i} \mid w_{l_{i-n+1},p_{i-n+1}} ... w_{l_{i-1},p_{i-1}}) \approx \prod_{i=1}^{m} p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}). \quad (2)$$

Actually, there are two hypotheses implied by the Markov hypothesis:

1. The limited history hypothesis: the probability of current word is dependent only on the previous *n-1* words, but irrelevant to the whole history of words.
2. The stationary hypothesis: the word transition probability is determined only by the words which consist of the transition probability, but irrelevant to the positions where these words possess in the sequence.

Formula (1) is firstly simplified by the limited history hypothesis, resulted in the second item of formula (2). Then, the stationary hypothesis is applied on it and the final form of the ngram model is obtained, as represented by the last item of formula (2). The paper substitutes w_{l_i} for w_{l_i,p_j} since the conditional probability is irrelevant to word position. In literature, the limited history hypothesis is referred to frequently, but seldom is the stationary hypothesis.

The most obvious way to extend the ngram model is simply to relax the limited history hypothesis and involve more history information of words. The higher-order ngram model is built up. However, the high-order ngram model suffers from the curse of dimensionality. As the model order increases, the parameter space explodes at an exponential rate. The data sparseness problem becomes very severe which hampers its applications gravely. From another point of view, the paper relaxes the stationary hypothesis and enhances the ngram model by the exploitation of the word positional information. The NS ngram model is proposed. It is described in the following sections.

3.2 NS Ngram Model

As presented in section 1, the occurrence of words is relevant to their positional information in sentence. It is beneficial for the language model to exploit the positional information to determine the word probability. However, the Markov hypothesis is too restricted to exploit the positional information due to its stationary assumption. The paper relaxes the stationary hypothesis of the traditional ngram model and proposes a non-stationary ngram model. The NS ngram model is formulized in as below:

$$P(s) \approx \prod_{i=1}^{m} p(w_{l_i}, p_i \mid w_{l_{i-n+1}}, p_{i-n+1} \ldots w_{l_{i-1}}, p_{i-1}) = \prod_{i=1}^{m} p(w_{l_i} \mid w_{l_{i-n+1}} \ldots w_{l_{i-1}}, t). \quad (3)$$

In the NS ngram model, formula (1) is simplified merely by the limited history hypothesis, rather than the stationary hypothesis. The conditional probability of the current word is determined not only by history words but also by the words' positions in sentence. The paper uses a single positional variable of t to denote the word positional information in formula (3). The traditional ngram model is a special case of the NS ngram model in which t is a constant.

Important things for the NS ngram model are how to calculate the value of t and how to estimate the conditional probability of word in formula (3).

3.3 Representation of *t*

Since t denotes the word positional information in a sentence, it is a natural way to take the word position index as the concrete value of t. However, there are two serious problems with this method. Firstly, index has different meanings in sentences of different lengths. For example, there are two English sentences: "Yesterday I saw you" and "Yesterday I saw you were looking around here". In both of the sentences, the word "you" has the same position index - 4. However, "you" appears at the end of the first sentence, while it is in the middle in the second. It possesses completely different positional information in these two sentences. Secondly, since a sentence may have arbitrary length, the t value can be any natural number. But computer can not deal with infinite value.

A refined method is to use the ratio of the word position index to the sentence length, which maps t into a real number in the range of [0, 1]. But there are infinite real numbers in that range and it can not make statistics based on each real number.

This paper divides the above range into several equivalent classes (bins). It assumes that the words in each bin share the same positional information. The value of t is set to the index of the according class. More formally, the above procedures are described as below:

1. Calculate the ratio of the word position index to the sentence's length, which maps t into the range of [0, 1].
2. Divide the range into several bins. The words in each bin share the same positional information.
3. Set the t value of current word as the index of the according bin.

Figure 1 shows an example of the above procedures:

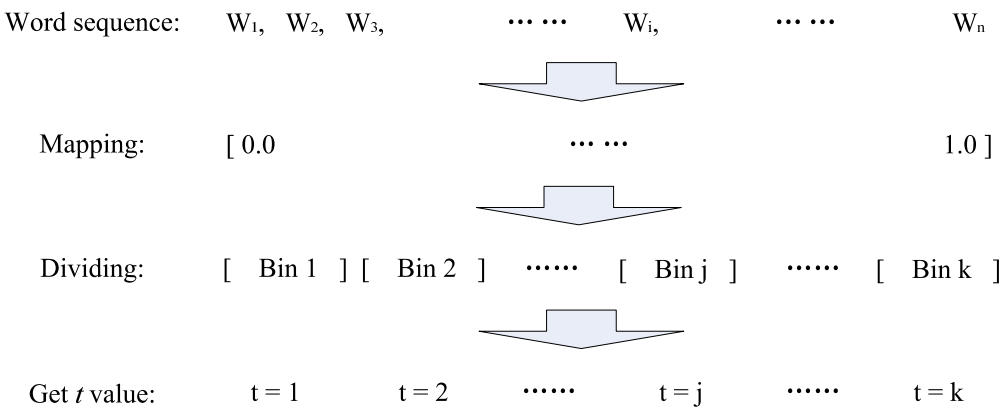


Figure 1. Calculation of the t value in NS ngram model

From the above procedures, the more number of bins it divides of the word sequence, the more accuracy of the positional information is extracted from the sentence.

3.4 Training Method

The section discusses how to estimate the conditional probability in formula (3), which is the training problem of the NS ngram model. Based on the representation of t in section 3.3, the sentences in the training corpus are divided into the same number of bins. The words in each bin share the same value of t. The paper builds up a specific ngram model for each value of t within each bin. All these specific ngram models constitute of the NS ngram model. Using k to denote the number of bins, there are totally k specific ngram models in the NS ngram model with k bins. The conditional probability of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$ is estimated under the Maximum Likelihood Estimation (MLE) principle:

$$p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t) = \frac{C(w_{l_{i-n+1}} ... w_{l_i}, t)}{C(w_{l_{i-n+1}} ... w_{l_{i-1}}, t)}. \quad (4)$$

$C(w_{l_{i-n+1}} ... w_{l_i}, t)$ is the occurrence times that the word sequence $w_{l_{i-n+1}} ... w_{l_i}$ falls in the t^{th} bin of the sentences in the training corpus. It is similar to interpreting $C(w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$.

In order to calculate the probability of a sentence, the t value is firstly obtained for each word. Then, the conditional probability of word is computed according to formula (4). Finally, the sentence probability is calculated by formula (3). The traditional ngram model is a special case of the NS ngram model in which there is only one bin.

4. Smoothing Techniques

As shown in section 3.4, there are totally k traditional ngram models in the NS ngram model with k bins. The space complexity of the NS ngram model is consequently k times more than

the traditional ngram model. Data sparseness problem is an inherent and severe problem in the traditional ngram model [Brown *et al*. 1992]. Therefore, it is more severe in the NS ngram model. Figure 2 illustrates the data sparseness problem in the NS ngram model.

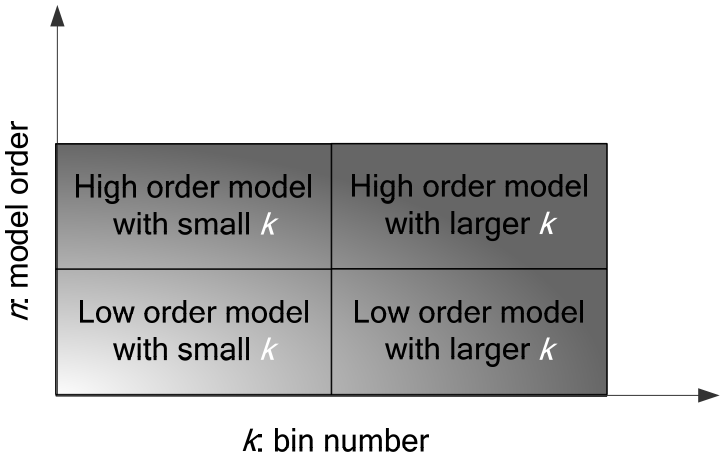


Figure 2. Data sparseness problem in NS ngram model

In Figure 2, the color of deep shade indicates that the data sparseness problem is severe in the NS ngram model, while the color of light shade means that the problem is not severe. As shown in Figure 2, there are two main factors in determining the degree of the data sparseness problem in the NS ngram model. They are the model order n and the bin number k. As n (or k) increases, the problem becomes more severe, and the estimated probability becomes more unreliable.

It is necessary to start with these two factors to solve the data sparseness problem of the NS ngram model. Considering the factor of the model order which is represented as the vertical axis in Figure 2, the high-order NS ngram model can be smoothed by lower-order NS ngram model, just as the traditional smoothing techniques do. It is our first smoothing approach. Considering the factor of the bin number which is shown as the horizontal axis, there are two ways to design the smoothing methods. The first way, the NS ngram model with larger value of k can be smoothed by the NS ngram model with smaller value of k. In particular, the traditional ngram model ($k=1$) can be utilized to smooth the NS ngram model ($k>1$). It is our second smoothing approach. The second way, the paper builds up a more compact form of the NS ngram model. It firstly constructs some statistical variables of the word positional information from the bins of the NS ngram model. Then, it calculates a weight from these variables for the traditional ngram probability. The weight is used to substitute for the concrete positional information which tends to cause the data sparseness problem in the NS ngram model. It is our third approach to smooth the NS ngram model. Until now, three smoothing approaches have been provided in sketch. They will be described in the following

sections in detail.

4.1 The First Approach

Since the NS ngram model is composed of several traditional ngram models, each of these component ngram models can be smoothed separately by the traditional smoothing techniques. The traditional smoothing techniques have been well studied before. Many smoothing algorithms have been proposed, such as the additive smoothing [Jeffreys 1948], the Good-Turing smoothing [Good 1953], the back-off smoothing [Katz 1987], the linear interpolation smoothing [Jelinek and Mercer 1980], the Kneser-Ney smoothing [Kneser and Ney 1995], and so on. Generally, they smooth the unreliable probabilities in the high-order ngram model by the reliable probabilities in the low-order ngram model. The paper can not try each existent smoothing algorithm on the NS ngram model. Three popular algorithms are taken in the paper. They are the additive smoothing, the back-off smoothing and the linear interpolation smoothing. The NS bigram model is taken as an example and the formulas are listed as below.

Additive smoothing:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \frac{C(w_{l_{i-1}}, w_{l_i}, t) + 1}{C(w_{l_{i-1}}, t) + l} \tag{5}$$

t is the positional variable which is defined in section 3.3; l is the lexicon size; and $\tilde{p}$ is the smoothed probability of the NS bigram model.

Back-off smoothing:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \begin{cases} P_{GT}(w_{l_i} \mid w_{l_{i-1}}, t) & \text{if } C(w_{l_{i-1}}, w_{l_i}, t) > 0 \\ \alpha(w_{l_{i-1}}, t)\, \tilde{P}(w_{l_i}, t) & \text{otherwise} \end{cases} \tag{6}$$

P_{GT} is the probability of the NS bigram model which is smoothed by the Good-Turing method. It is formalized as below:

$$P_{GT}(w_{l_i} \mid w_{l_{i-1}}, t) = \frac{C_{GT}(w_{l_{i-1}}, w_{l_i}, t)}{C(w_{l_{i-1}}, t)} \tag{7}$$

and

$$C_{GT}(w_{l_{i-1}}, w_{l_i}, t) = (C(w_{l_{i-1}}, w_{l_i}, t) + 1) \times \frac{E(C(w_{l_{i-1}}, w_{l_i}, t) + 1)}{E(C(w_{l_{i-1}}, w_{l_i}, t))} \tag{8}$$

E(C) is the expectation of the number of the bigram items which occurs C times in the corpus. In reality, *N(C)* is usually substituted for *E(C)*. *N(C)* is the concrete number of the bigram

items which actually occurs C times in the training corpus. Formula (8) is reformulated as below:

$$C_{GT}(w_{l_{i-1}}, w_{l_i}, t) = (C(w_{l_{i-1}}, w_{l_i}, t) + 1) \times \frac{N(C(w_{l_{i-1}}, w_{l_i}, t) + 1)}{N(C(w_{l_{i-1}}, w_{l_i}, t))} \tag{9}$$

However, $N(C)$ can not be estimated reliably for some large values of C. At this time, formula (9) can not work properly and problems occur in the Good-Turing method. In particular, when C reaches its max value in the training corpus, $C_{GT}(w_{l_{i-1}}, w_{l_i}, t)$ is calculated to be zero according to formula (9) because $N(C+1)$ is equal to zero. It is obviously wrong. In this paper, a simple strategy is adopted to address the problem. Formula (7) and formula (9) are adopted only for the small value of C (i.e. below a threshold). For the large value of C, it is regarded that the bigram probabilities can be estimated reliably according to the word frequencies and they need not to be smoothed. The MLE principle is applied on them directly.

In formula (6), α is the coefficient for normalization and it is calculated as below:

$$\alpha(w_{l_{i-1}}, t) = \frac{\beta(w_{l_{i-1}}, t)}{\sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)=0} \tilde{P}(w_{l_i}, t)} = \frac{\beta(w_{l_{i-1}}, t)}{1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)>0} \tilde{P}(w_{l_i}, t)} \tag{10}$$

and

$$\beta(w_{l_{i-1}}, t) = 1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)>0} P_{GT}(w_{l_i} \mid w_{l_{i-1}}, t) \tag{11}$$

Linear interpolation smoothing:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \lambda(t) \times P(w_{l_i} \mid w_{l_{i-1}}, t) + (1 - \lambda(t)) \times \tilde{P}(w_{l_i}, t) \tag{12}$$

P is the probability of the NS bigram model which is estimated by formula (4); $\lambda(t)$ is the coefficient which is a function of t and can be estimated by the EM algorithm on the held-out corpus.

4.2 The Second Approach

As shown in Figure 2, when the value of k increases, there are more probability distributions in the NS ngram model to be estimated on the training corpus. The conditional probability becomes more specific and unreliable, and the data sparseness problem of the NS ngram model becomes more severe. Usually, the smoothing techniques utilize the general and reliable probability distributions to smooth the specific and unreliable ones. Therefore, it can make use of the reliable probability of the NS ngram model with small k, to smooth the unreliable probability of the NS model with large k. In particular, it can utilize the traditional

ngram model (*k=1*) to smooth the NS ngram model (*k>1*). However, the traditional ngram model also suffers from the data sparseness problem. Actually, the paper utilizes the smoothed traditional ngram model in this approach.

Totally, three smoothing methods are investigated. They are the back-off method, the linear interpolation method and the hybrid method. The formulas are listed as below.

Back-off smoothing:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \begin{cases} P_{GT}(w_{l_i} \mid w_{l_{i-1}}, t) & \text{if } C(w_{l_{i-1}} w_{l_i}, t) > 0 \\ \alpha^1(w_{l_{i-1}}, t)\,\tilde{P}(w_{l_i} \mid w_{l_{i-1}}) & \text{otherwise} \end{cases} \tag{13}$$

α^1 is the coefficient for normalization, and it can be calculated as below:

$$\alpha^1(w_{l_{i-1}}, t) = \frac{\beta^1(w_{l_{i-1}}, t)}{\sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)=0} \tilde{P}(w_{l_i} \mid w_{l_{i-1}})} = \frac{\beta^1(w_{l_{i-1}}, t)}{1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)>0} \tilde{P}(w_{l_i} \mid w_{l_{i-1}})} \tag{14}$$

and

$$\beta^1(w_{l_{i-1}}, t) = 1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i}, t)>0} P_{GT}(w_{l_i} \mid w_{l_{i-1}}, t) \tag{15}$$

In formula (13), $\tilde{P}(w_{l_i} \mid w_{l_{i-1}})$ is the traditional bigram probability smoothed by the back-off method, and it is calculated as below:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}) = \begin{cases} P_{GT}(w_{l_i} \mid w_{l_{i-1}}) & \text{if } C(w_{l_{i-1}} w_{l_i}) > 0 \\ \alpha^2(w_{l_{i-1}})\,\tilde{P}(w_{l_i}) & \text{otherwise} \end{cases} \tag{16}$$

α^2 is the coefficient for normalization, and it can be computed as below:

$$\alpha^2(w_{l_{i-1}}) = \frac{\beta^2(w_{l_{i-1}})}{\sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i})=0} \tilde{P}(w_{l_i})} = \frac{\beta^2(w_{l_{i-1}})}{1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i})>0} \tilde{P}(w_{l_i})} \tag{17}$$

and

$$\beta^2(w_{l_{i-1}}) = 1 - \sum_{w_{l_i}: C(w_{l_{i-1}} w_{l_i})>0} P_{GT}(w_{l_i} \mid w_{l_{i-1}}) \tag{18}$$

Linear interpolation smoothing:

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \lambda(t) \times P(w_{l_i} \mid w_{l_{i-1}}, t) + (1 - \lambda(t)) \times \tilde{P}(w_{l_i} \mid w_{l_{i-1}}) \tag{19}$$

$\tilde{P}(w_{l_i} \mid w_{l_{i-1}})$ is the traditional bigram probability smoothed by the linear interpolation method, and it is calculated by formula (20):

$$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}) = \theta \times P(w_{l_i} \mid w_{l_{i-1}}) + (1-\theta) \times P(w_{l_i}) \tag{20}$$

The coefficients of $\lambda(t)$ and θ can be optimized by the EM algorithm on the held-out corpus.

Hybrid smoothing:

$$\hat{P}(w_{l_i} \mid w_{l_{i-1}}, t) = \lambda(t) \times \tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t) + (1-\lambda(t)) \times \tilde{P}(w_{l_i} \mid w_{l_{i-1}}) \tag{21}$$

$\tilde{P}(w_{l_i} \mid w_{l_{i-1}}, t)$ is the NS bigram probability smoothed by the back-off method, and it can be calculated by formula (6); $\tilde{P}(w_{l_i} \mid w_{l_{i-1}})$ is the traditional bigram probability smoothed by the back-off method, and it can be calculated by formula (16). These two probabilities are interpolated into a hybrid probability of $\hat{P}(w_{l_i} \mid w_{l_{i-1}}, t)$ which forms the hybrid smoothing method.

4.3 The Third Approach

The above sections provide two smoothing approaches for the NS ngram model. They are mainly based on the traditional smoothing techniques. This section proposes a novel smoothing method and constructs a more compact model to solve the data sparseness problem of the NS ngram model.

As shown in Figure 2, the model order and the bin number are two main factors in determining the degree of the data sparseness problem in the NS ngram model. The first one is also the dominant factor of the traditional ngram model. Then, the data sparseness in the NS ngram model, which is brought forth by the first factor, can be regarded as inheriting from the traditional ngram model. The second factor is specific to the NS ngram model. It brings forth the data sparseness problem when the positional information is modeled. Based on the above analysis, the smoothing method for the NS ngram model can be decomposed into two steps. The first step is to solve the data sparseness problem which is brought forth by modeling the word positional information. Some statistical variables are constructed to substitute for the concrete positional information. A more compact model is built up. The second step is to solve the data sparseness problem which is inherited from the traditional ngram model. The traditional smoothing techniques are utilized.

After describing the motivation and the technique sketch, the formula is presented as below:

$$\tilde{p}(w_{l_i} \mid w_{l_{i-1}}, t) = \frac{1}{Z(w_{l_{i-1}})} e^{\frac{\alpha \times V(w_{l_i})}{((t-E(w_{l_i}))^2+\beta)}} \times \tilde{p}(w_{l_i} \mid w_{l_{i-1}}) \quad (22)$$

where

- t is the positional variable.
- $E(w_{l_i})$ is the expectation of the positional information of w_{l_i} in the training corpus.
- $V(w_{l_i})$ is the variance of the positional information of w_{l_i} in the training corpus.
- α and β are the coefficients to adjust the weight.
- $\tilde{p}(w_{l_i} \mid w_{l_{i-1}})$ is the smoothed traditional bigram probability. Any smoothing algorithm, such as the back-off algorithm and the linear interpolation algorithm, can be applied.
- $Z(w_{l_{i-1}})$ is the factor for normalization and it is defined as below:

$$Z(w_{l_{i-1}}) = \sum_{l_i=1}^{l_i=l} e^{\frac{\alpha \times V(w_{l_i})}{((t-E(w_{l_i}))^2+\beta)}} \times \tilde{p}(w_{l_i} \mid w_{l_{i-1}}) \quad (23)$$

- l is the size of the lexicon

To smooth the word positional information, the paper aims at reducing the parameter number of the NS ngram model. Different from the clustering technique in the class-based ngram model [Brown *et al*. 1992], the paper constructs the statistical variables of the word positional information to substitute for the concrete value of t in the NS ngram model. Two statistical variables are calculated: the expectation and the variance. The weight is computed for the bigram probability according to these variables. Such an assumption is made that more weight should be awarded if the current word position fits in better with the training corpus, and less weight vice versa. According to the assumption, the term of $t\text{-}E(w_{li})$, which defines the difference between the current word position and its average position in the training corpus, is adopted in formula (22). As the value decreases, t fits in with the training corpus better and more weight should be awarded. Henceforth, the weight function is descendent with the value of $t\text{-}E(w_{li})$ as formula (22) shows. Moreover, the weight function is ascendant with the variance $V(w_{li})$. The term $V(w_{li})$ is mainly used to balance the value of the term $t\text{-}E(w_{li})$ for some *active* words. For example, some adjectives can appear at any position in a sentence. Then it is unreasonable to decrease the weight just as the term $t\text{-}E(w_{li})$ increases. In such a situation, the value of $V(w_{li})$ of the *active* word is usually bigger than that of the *inactive*. Then it can provide a balance for the value of $t\text{-}E(w_{li})$. Until now, the section has described the method to solve the data sparseness problem which is brought forth by modeling the word positional information. It is the first step of this approach to smooth the NS ngram model. It

should be noticed that the way to constructing the weight is a purely empirical method. There is no theoretic foundation on it. However, it performs pretty well in the experiments, as presented later in section 5.4.3. In the second step, the traditional smoothing techniques can be adopted to solve the data sparseness problem which inherits from the traditional ngram model. The paper investigates two smoothing techniques: the back-off smoothing and the linear interpolation smoothing.

Moreover, the coefficients of α and β can be optimized by some automatic methods on the held-out corpus. The genetic algorithm is adopted in this paper. It is presented as below:

Algorithms: Genetic algorithm to optimize α and β

Input: The held-out corpus

Output: The optimal value of α and β

---

1. Initiation: generate the initial population of α and β randomly
2. Evolution of population
   - Step 1: calculate fitness for each individual
   - Step 2: selection
   - Step 3: crossover
   - Step 4: mutation
   - Step 5: if termination criterion is met
     - go to 3
     - else
     - go to step 1
3. Choose the best individual as the solution

The actual performance of formula (22) on the held-out corpus is taken as the fitness function in the above algorithm.

Until now, a compact NS ngram model has been built up in the section. The parameter space is reduced by substituting the statistical variables for the concrete positional information, which results in a space complexity of $O(l^n+2l+2)$. The data sparseness problem is alleviated. However, the predictive capability is also lowered to some extent due to the small parameter space, which is the limitation of this smoothing approach. To overcome the above drawback, the paper constructs the statistical variables for the word ngram other than for the word itself. It results in a larger space complexity of $O(3\times l^n)$, and therefore yields a more powerful predictive capability. In addition, the compact model has a slight higher time complexity than the normal NS ngram model by calculation of the weight function.

5. Experiments and Discussions

This section evaluates the NS ngram model and its smoothing techniques on the pinyin-to-character conversion task which is the core technique of the Chinese keyboard input method. The section is organized as follows. Firstly, the task and the data set are described. Secondly, the non-stationary property of words is investigated in a statistical way so as to verify the motivation of the paper. Thirdly, the performance of the NS ngram model is presented and compared with the traditional ngram model. Finally, the smoothing algorithms proposed in the paper are evaluated and the performances of the smoothed NS ngram model are provided.

5.1 Task and Data Set Description

Task Description

The standard keyboard is initially designed for native English speakers. In Asia, such as China, Japan and Thailand, people can not input their language through the standard keyboard directly. Asian text input becomes the challenge for the computer users in Asia. Asian language input method is one of the most important techniques in Asian language processing. The pinyin-based input method is the most important Chinese text input method. There are over 97% of Chinese computer users using pinyin to input Chinese text [Chen 1997]. According to the scale of the input unit, the pinyin-based input method can be categorized into three types: the character-level input method, the word-level or phrase-level input method and the sentence-level input method respectively. The sentence-level input method becomes the most prevalent pinyin-based input method due to its high precision. The pinyin-to-character conversion task aims to convert the sequence of pinyin strings into one Chinese sentence. It is the core technique of the sentence-level pinyin-based Chinese text input method. Therefore, the improvement on the pinyin-to-character conversion task has a great effect on Chinese text input method.

In Chinese, there are totally 410 pinyin symbols (without the tone information) which correspond to more than 30,000 Chinese characters. For a certain inputted pinyin sequence, there are many candidates of Chinese character sequence corresponding to it, but only one is what the user really wants to obtain. Language model is to select the most probable one among these candidates. Error rate is usually used to evaluate the performance of a language model on this task.

The pinyin-to-character conversion task can also be taken as a simplified automatically speech recognition task [Gao *et al*. 2005]. Both of the two tasks aim to convert the phonetic information into the character sequence. However, unlike the speech recognition task, the pinyin-to-character conversion task doesn't have to deal with the acoustic ambiguity because

the pinyin strings are directly inputted on the keyboard by user. Therefore, our techniques also illuminate to the speech recognition task.

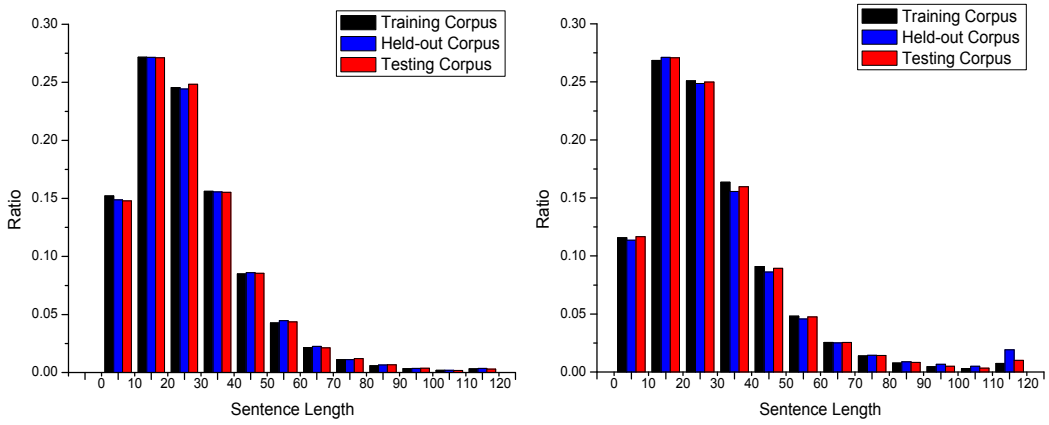
Text Corpus

The paper chooses the 6763 Chinese frequent characters as lexicon. Two sets of the People's Daily corpus are adopted in the experiments: the half year of corpus in 1998 for the experiments of the NS bigram model and the whole year of corpus in 2000 for the experiments of the NS trigram model. Each set of corpus is divided into three parts: the training corpus, the held-out corpus and the testing corpus. The detailed information is listed in Table 1.

Table 1. Description of text corpus

	Training (months / #characters)	**Held-out** (months / #characters)	**Testing** (months / #characters)
People's Daily corpus in 1998	1-5 months 9.09×10^6	1/3 of 6th month 6.29×10^5	2/3 of 6th month 1.25×10^6
People's Daily corpus in 2000	1-11 months 2.27×10^7	1/3 of 12th month 7.01×10^5	2/3 of 12th month 1.40×10^6

The paper chooses the large scale of corpus for the NS trigram model since its parameter space is much larger than that of the NS bigram model. In what follows, the paper presents the distributions of the lengths of the sentences in those corpora. The information is crucial to evaluating the NS ngram model which exploits the positional information of word in the sentences. The distributions are presented in Figure 3.



a. .Distributions of Sentence Length on 1998 Corpus

b. .Distributions of Sentence Length on 2000 Corpus

Figure 3. Distributions of the sentence length in text corpus

According to Figure 3, most of the lengths of the sentences fall in the range from 10 to 60. The average lengths of sentences are 27.41 on the corpus in 1998, and 29.64 on the corpus in 2000 respectively. Moreover, the distributions of the sentences' lengths are much similar to each other among the three parts of the text corpus.

Pinyin Corpus

The pinyin corpus is necessary for evaluating the NS ngram models on the pinyin-to-character conversion task. The paper gets the pinyin corpus from the above text corpus by a conversion toolkit[1] which yields 99.7% accuracy evaluated on a golden corpus. When the NS ngram models are evaluated, the pinyin corpus is firstly converted into the text corpus by the NS ngram model. Then, the converted results are compared with the standard text corpus and the error rate is calculated. As the pinyin corpus is not a golden corpus, the errors in the pinyin corpus could lead to the conversion error of the NS ngram model. Therefore, the actual error rate of the NS ngram model is a little lower than the reported results in the paper and the NS ngram model could get a little better performance in the real system. However, since there are not many errors in the pinyin corpus because of the high precision of the conversion toolkit, the reported error rate of the NS ngram model can be regarded to be close enough to the actual error rate.

5.2 Non-Stationary Property of Words

Section 1 has provided some intuitive examples for the non-stationary property (NS property) of words. However, the intuition is not enough for our motivation of the paper. The section will further present some statistical evidences.

The NS property assumes that word behaves differently in different portions of sentences. Then their probability distributions would be different in different portions. The more differences between these distributions, the more positional information has been implied by word. The section investigates the probability distributions in the NS bigram model, and presents their differences by comparing them with the distribution in the traditional bigram model. The Kullback-Leibler (KL) distance [Cover and Thomas 1991] is taken as the metric. And only if the distances are great enough, could the NS bigram model be expected to outperform the traditional bigram model; otherwise, they would have similar performances.

As mentioned in section 3, there are totally k probability distributions in the NS ngram model with k bins. So there are k different KL distances to be calculated between the traditional bigram model and the NS bigram model. The section calculates these KL distances

[1] The toolkit can be obtained freely from the link:
http://www.insun.hit.edu.cn/product/viewproduct.asp?id=105

for the NS bigram model with different k values. The experimental results are summarized in Table 2.

Table 2. The KL distances between the traditional bigram model and the NS bigram model

Bin number / Bin index	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
t=1	0	0.11	0.15	0.19	0.24	0.28	0.32	0.37
t=2	---	0.05	0.08	0.08	0.10	0.11	0.12	0.14
t=3	---	---	0.13	0.09	0.09	0.09	0.09	0.10
t=4	---	---	---	0.21	0.10	0.09	0.09	0.09
t=5	---	---	---	---	0.32	0.12	0.10	0.09
t=6	---	---	---	---	---	0.42	0.13	0.10
t=7	---	---	---	---	---	---	0.52	0.14
t=8	---	---	---	---	---	---	---	0.62
Average KL Distance	0	0.08	0.12	0.15	0.17	0.18	0.19	0.21

In the row of Table 2, the section lists the NS bigram models with various values of k which are up to 8. In the column, it calculates the KL distance between each distribution of the NS bigram model and the distribution of the traditional bigram model. At last, it calculates the average KL distance for each NS bigram model.

According to the experimental results in Table 2, it is found that as k increases, the average KL distance becomes larger and larger, indicating that there are more and more differences between the distributions of the NS bigram model and that of the traditional bigram model. Therefore, more and more positional information is modeled by the NS bigram model, and more predictive capability is expected. Moreover, focusing on a certain column in Table 2, i.e. the column of $k=5$, it calculates the KL distance for each distribution of the NS bigram model with *5* bins. It is found that the KL distances calculated from the marginal positions are greater than the distances from the middle ones. For example, the KL distances of *t=1 (0.24)* and *t=5 (0.32)* are greater than the distance of *t=3 (0.09)*. It is more obvious for the lager value of k. It indicates that the distributions in the marginal positions represent more positional information, and therefore contribute more to the ultimate performance of the NS bigram model than the middle ones.

5.3 Experiments of NS Ngram Model

This section evaluates the un-smoothed NS ngram model on the pinyin-to-character conversion task. Two sets of experiments, the close test and the open test, are carried out. The

test on the training corpus is referred to as the close test; and the test on the testing corpus is referred to as the open test. In order to avoid the zero-probability problem in the open test, the paper adds a small value[2] to the zero-frequency words when estimating their probabilities. The un-smoothed traditional ngram model is taken as the baseline model. Both the NS bigram model and the NS trigram model are investigated. The experimental results of the NS bigram model are firstly presented in Table 3.

Table 3. Experimental results of the NS bigram model

Bin Number		*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
Close test	**Error Rate**	8.30%	7.17%	6.55%	6.08%	5.74%	5.43%	5.19%	**4.98%**
	Reduction	---	13.61%	21.08%	26.75%	30.84%	34.58%	37.47%	**40.00%**
Open test	**Error Rate**	14.97%	**12.62%**	13.16%	13.61%	13.93%	14.23%	14.52%	14.81%
	Reduction	**---**	**15.70%**	12.09%	9.08%	6.95%	4.94%	3.01%	1.07%

As mentioned in section 3.4, the traditional bigram model can be regarded as the NS bigram model in which *k=1*. According to the experimental results in Table 3, the NS bigram model outperforms the traditional bigram model significantly. It yields as much as 40% error rate reduction in the close test, and 15.7% reduction in the open test. It proves that the NS bigram model has more powerful predictive capability than the traditional bigram model. Moreover, as the value of *k* increases, the error rate of the NS bigram model in the close test is reduced constantly, proving that the improvement of the NS ngram model is due to the increasing positional information of word. However, in the open test, the error rate stops decreasing after *k=2*, because the data sparseness problem becomes more severe as *k* increases.

The NS trigram model is also investigated. The experimental results are presented in Table 4.

Table 4. Experimental results of the NS trigram model

Bin Number		*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
Close test	**Error Rate**	2.21%	1.80%	1.73%	1.65%	1.61%	1.59%	1.57%	**1.57%**
	Reduction	---	18.55%	21.71%	25.34%	27.15%	28.05%	28.96%	**28.96%**
Open test	**Error Rate**	**18.92%**	19.72%	20.55%	21.34%	21.94%	22.61%	23.22%	23.74%
	Reduction	**---**	-4.06%	-8.61%	-12.79%	-15.96%	-19.50%	-22.72%	-25.47%

[2] It is the minimum positive floating point value in the Windows system (the DBL_MIN constant), and has the value of 2.22×10^{-308}.

The experimental results are similar to those of the NS bigram model. As presented in Table 4, the NS trigram model outperforms the traditional trigram model significantly in the close test, and has achieved as much as 28.96% error rate reduction. It proves that the NS trigram model is more powerful than the traditional trigram model. Moreover, the error rate decreases along with the k value, proving that the improvements of the NS trigram model are due to the increasing positional information of word. However, unlike the NS bigram model, the NS trigram model performs worse in the open test, indicating that the NS trigram model suffers from much more severe data sparseness problem than the NS bigram model even though a larger training corpus is adopted in the experiments.

To sum up, the NS ngram model achieves great improvements by exploiting the word positional information; however, it suffers from severe data sparseness problem. The following sections will investigate the smoothing techniques presented in section 4, and provide the experimental results of the smoothed NS ngram model. Without loss of the generality, all the following experiments are carried out on the NS bigram model.

5.4 Experiments of Smoothing Techniques

This section firstly investigates the three smoothing approaches separately. Then, these techniques are compared to each other and some conclusions are drawn. Finally, it investigates the performance of each probability distribution of the smoothed NS bigram model so as to gain further insight. All the experiments are carried out in the open test since the data sparseness problem occurs only on the unseen data.

5.4.1 The First Approach

This approach smoothes the probability distributions in the NS bigram model by the traditional smoothing techniques. Totally three smoothing algorithms are investigated: the additive smoothing, the back-off smoothing and the linear interpolation smoothing. The techniques have been well presented in section 4.1. The un-smoothed NS bigram model is taken as the baseline model from which the error rate reduction is calculated. The experimental results are provided in Table 5.

Firstly, according to the experimental results, the traditional smoothing techniques smooth the NS bigram model effectively. It yields great error rate reductions on the pinyin-to-character conversion task. For example, as much as 15.77% error rate reduction has been yielded by the back-off smoothing technique. Secondly, the error reductions of the smoothed NS bigram model become more significant when $k>2$. It indicates that as the value of k increases, the data sparseness problem becomes more and more severe, and the smoothing technique plays a more important role. However, the most significant error rate reduction occurs at $k=1$ which is the traditional bigram model. It is for the reason that the baseline

accuracy of the traditional bigram model is relative lower than those of the NS bigram models. Thirdly, the error rate of the smoothed NS bigram model still increases when $k>2$, just as the un-smoothed NS bigram model does. It proves that the NS bigram model smoothed by this approach can not make full use of the increasing positional information of word so as to gain further improvements. It indicates that this smoothing approach can only *alleviate* the data sparseness problem of the NS bigram model, but can not really *solve* it.

Table 5. Experimental results of the first smoothing approach

Bin Number		*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
Un-smoothed	**Error Rate**	14.97%	**12.62%**	13.16%	13.61%	13.93%	14.23%	14.52%	14.81%
Additive	**Error Rate**	13.63%	**12.22%**	12.58%	12.9%	13.12%	13.41%	13.61%	13.87%
	Reduction	8.95%	3.17%	4.41%	5.22%	5.81%	5.76%	6.27%	**6.35%**
Back-off	**Error Rate**	12.4%	**10.88%**	11.24%	11.54%	11.78%	12.05%	12.23%	12.51%
	Reduction	17.17%	13.79%	14.58%	15.21%	15.43%	15.32%	**15.77%**	15.53%
Interpolation	**Error Rate**	12.17%	**11.00%**	11.42%	11.79%	12.07%	12.35%	12.58%	12.86%
	Reduction	18.7%	12.84%	13.22%	13.37%	13.35%	13.21%	**13.50%**	13.17%

5.4.2 The Second Approach

In the second approach, the paper smoothes the NS bigram model by the traditional bigram model. Three smoothing algorithms are provided. They are the back-off method, the linear interpolation method and the hybrid method, as described in section 4.2. The experimental results are presented in Table 6.

Table 6. Experimental results of the second smoothing approach

Bin Number		*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
Un-smoothed	**Error Rate**	14.97%	**12.62%**	13.16%	13.61%	13.93%	14.23%	14.52%	14.81%
Back-off	**Error Rate**	12.4%	**10.54%**	10.83%	11.16%	11.47%	11.83%	12.18%	12.49%
	Reduction	17.17%	16.48%	17.71	18%	17.66%	**16.87%**	16.12%	15.67%
Interpolation	**Error Rate**	12.17%	10.46%	10.46%	10.44%	10.4%	10.37%	**10.36%**	10.37%
	Reduction	18.7%	17.12%	20.52%	23.29%	25.34%	27.13%	28.65%	**29.98%**
hybrid	**Error Rate**	12.4%	10.42%	10.34%	10.27%	10.21%	10.16%	**10.12%**	10.13%
	Reduction	17.17%	17.43%	21.43%	24.54%	26.70%	28.60%	30.30%	**31.80%**

According to the experimental results, the second smoothing approach is more effective in smoothing the NS bigram model than the first one. For example, the hybrid method yields as much as 31.8% error rate reduction which is much higher than the best result of the first smoothing approach (which is 15.77% yielded by the back-off method). Moreover, for the linear interpolation method and the hybrid method, the error rate of the smoothed NS bigram model no longer increases along with the k value as the un-smoothed NS bigram model does, but decreases constantly. It proves that the NS bigram model smoothed by these methods can make full use of the increasing positional information of word and get further improvements. It can be concluded that these smoothing methods can really solve the data sparseness problem of the NS bigram model, rather than just alleviate the problem. The back-off smoothing method does not perform as well as the above two methods because it is based on the model selection methodology and can not make full use of each component model.

5.4.3 The Third Approach

The third approach smoothes the NS bigram model by reducing its parameter space and building up a more compact model. The statistical variables are utilized to substitute for the concrete positional information. A weight is calculated from these variables for the traditional bigram probability. The traditional smoothing techniques are utilized to smooth the bigram probability. Two smoothing techniques are investigated in the section: the back-off smoothing and the linear interpolation smoothing. The coefficients of α and β are optimized by the genetic algorithm on the held-out corpus. The settings of the genetic algorithm are presented in Table 7.

Table 7. Settings of the genetic algorithm

Population size	30
Probability of reproduction	0.1
Probability of crossover	0.65
Probability of mutation	0.2
Selection mechanism	Rank selection
Crossover mechanism	Arithmetical crossover
Mutation mechanism	Normal mutation
Fitness function	Error rate of the pinyin-to-character converter

The un-smoothed NS bigram model is taken as the baseline model. The experimental results are presented in Table 8.

Table 8. Experimental results of the third smoothing approach

Bin Number		*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
Un-smoothed	**Error Rate**	14.97%	**12.62%**	13.16%	13.61%	13.93%	14.23%	14.52%	14.81%
Back-off	**Error Rate**	12.4%	10.59%	10.47%	10.47%	10.43%	10.43%	10.43%	**10.41%**
	Reduction	17.17%	16.09%	20.44%	23.07%	25.13%	26.70%	26.70%	**29.71%**
Interpolation	**Error Rate**	12.17%	10.56%	10.48%	10.44%	10.43%	10.42%	10.43%	**10.4%**
	Reduction	18.7%	16.32%	20.36%	23.29%	25.13%	26.77%	28.17%	**29.78%**

Firstly, according to the experimental results, this approach can smooth the NS bigram model effectively. It achieves as much as 29.78% error rate reduction which is slightly lower than the second approach's (31.8%), whereas much higher than the first one's (15.77%). This smoothing approach can not achieve the best performance because the compact model has a smaller parameter space and its predictive capability is lower than that of the NS bigram model. Secondly, the error rate of the smoothed NS bigram model decreases along with the k value constantly. It proves that the approach can really solve the data sparseness problem of the NS bigram model, just as the second approach does. Finally, the performance of the smoothed NS bigram model becomes stably after $k=2$, which indicates that a small number of bins are enough to estimate the statistical variables and get the performance improvements.

5.4.4 Comparisons

This section compares the performances of the three smoothing approaches with each other. In each approach, it presents the smoothing algorithm which yields the best experimental results. The smoothed traditional bigram model is also presented for comparison. The results are summarized in Figure 4.

According to Figure 4, several conclusions can be drawn as follows. Firstly, the smoothed NS bigram model outperforms the smoothed traditional bigram model significantly by the exploitation of the word positional information. Secondly, all the smoothing approaches smooth the NS bigram model effectively with great error rate reduction. Thirdly, the second and the third approaches perform better than the first one. They can make full use of the positional information and really solve the data sparseness problem of the NS bigram model. Finally, the third approach yields the comparable experimental results with the second one, while it needs much smaller parameter space.

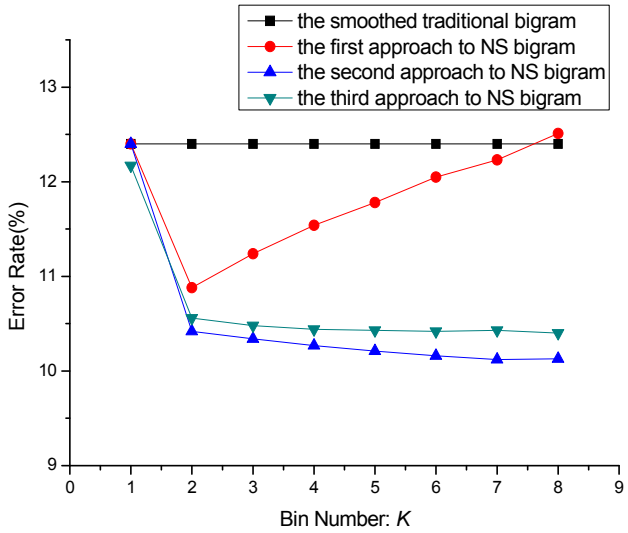


Figure 4. Comparison of the three smoothing approaches

5.4.5 Performance of Each Distribution in NS Bigram Model

In section 5.2, it has presented the NS property of words by investigating the probability distributions in the NS bigram model. In order to gain more insight, this section presents the performance of each probability distribution in the NS bigram model and evaluates their contributions to the ultimate performance of the NS bigram model.

Generally speaking, it can not tell exactly which probability distribution in the NS bigram model leads to a certain error in the pinyin-to-character conversion process. An approximate method is then provided. The section simply divides each sentence of the test corpus into several bins according to the method in section 3.3, and then calculates the error rate in each bin separately. Each error rate corresponds to the performance of a particular probability distribution in the NS bigram model. All the following experiments are carried out in the open test. The hybrid algorithm in the second approach is utilized to smooth the NS bigram model. It yields the best experimental results in the above sections. The NS bigram model is built up on various values of k which are up to 8. The experimental results are summarized in Table 9.

Table 9. Performance of each probability distribution in the NS bigram model

Bin number / Bin index	*k*=1	*k*=2	*k*=3	*k*=4	*k*=5	*k*=6	*k*=7	*k*=8
***t*=1**	12.4%	11.29%	11.06%	10.93%	10.70%	10.52%	10.42%	10.28%
***t*=2**	---	9.48%	11.46%	11.18%	11.05%	10.93%	10.85%	10.77%
***t*=3**	---	---	8.29%	11.39%	11.50%	11.20%	11.20%	10.99%
***t*=4**	---	---	---	7.14%	10.96%	11.45%	11.23%	11.24%
***t*=5**	---	---	---	---	6.13%	10.58%	11.17%	11.54%
***t*=6**	---	---	---	---	---	5.33%	10.18%	11.09%
***t*=7**	---	---	---	---	---	---	4.52%	9.62%
***t*=8**	---	---	---	---	---	---	---	3.81%
Overall error rate	12.4%	10.42%	10.34%	10.27%	10.21%	10.16%	10.12%	10.13%

In the row of Table 9, the section lists the NS bigram model with various values of k which are up to 8. In the column, it presents the error rate of each probability distribution of the NS bigram model. In the last line, it lists the overall error rate of the NS bigram model.

Focusing on a certain column in Table 9, the error rates of the probability distributions in the marginal positions are generally lower than those in the middle positions in the NS bigram model. For example, in the NS bigram model with *k=5*, the error rates of *t=1(10.7%)* and *t=5(6.13%)* are much lower than the error rate of *t=3 (11.5%)*. It is more obvious for the larger values of *k*. The experimental results verify our speculations in section 5.2 and prove that the distributions in the marginal positions have more predictive capabilities than the middle ones, and consequently contribute more to the ultimate performance of the NS bigram model. In addition, it is found that the error rate at the end position is much lower than those in other positions. In the above example, the error rate of *t=5(6.13%)* is much lower than others. It is because many of punctuations are modeled in this probability distribution. These punctuations, such as full stop and exclamation, always appear at the end of the sentence. Their positional information is much richer than words'. Therefore, the predictive capability of the probability distribution at the end position is much more powerful than other distributions in the NS bigram model, and it yields much higher performance.

6. Conclusions

This paper enhances the traditional ngram model by relaxing the stationary hypothesis and exploring the word positional information. The non-stationary ngram model is proposed. Several related issues are discussed in detail, including the definition of the NS ngram model,

the representation of the word positional information and the estimation of the conditional probability. In addition, three smoothing approaches are proposed to solve the data sparseness problem of the NS ngram model. Several smoothing algorithms are presented in each approach. In the experiments, the NS ngram model and its smoothing techniques are evaluated on the pinyin-to-character conversion task which is the core technique of Chinese text input method. According to the experimental results, several conclusions are drawn as follows:

1. The NS ngram model outperforms the traditional ngram model significantly by the exploitation of the word positional information; however, it suffers from severe data sparseness problem.
2. The traditional smoothing techniques are effective in smoothing the NS ngram model; however, they can only alleviate the data sparseness problem without solving it completely.
3. The traditional ngram model is utilized to smooth the NS ngram model. Combined with the traditional smoothing techniques, this smoothing approach can solve the data sparseness problem completely and achieve the best experimental results.
4. The third smoothing approach can also solve the data sparseness problem of the NS ngram model, and it yields a comparable experimental result to the second approach at the cost of a smaller parameter space.
5. Among the probability distributions in the NS ngram model, the distributions in the marginal positions have more predictive capability than the middle ones, and therefore contribute more to the ultimate performance of the NS ngram model.

Acknowledgments

This investigation was supported by the key project of the National Natural Science Foundation of China (“Research on Theory and Technique of Question-Answering Information Retrieval”, grant No.60435020), the project of the National Natural Science Foundation of China (“Research on the Non-stationary Property of Language Element in Natural Language Processing”, grant No.60673037), the project of the High Technology Research and Development Program of China (“Intelligent Search Engine based on Natural Language Processing”, grant No. 2006AA01Z197) and the project of MOE-MS Key Laboratory of Natural Language Processing and Speech in China (“Lexicon Construction for Statistical Language Modeling on Special Area”, grant No.01307620).

We especially thank the anonymous reviewers for their valuable suggestions and comments.

References

Brown, P. F., S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer, "The Mathematics of Statistical Machine Translation: Parameter Estimation," *Computational Linguistics*, 19(2), 1992, pp. 269-311.

Brown, P. F., V. J. D. Pietra, and P. V. deSouza, "Class-based n-gram models of natural language," *Computational Linguistics*, 18(4), 1992, pp. 467-479.

Carpenter, B., "Scaling high-order character language models to gigabytes," In *Proceedings of the Association for Computational Linguistics Software Workshop*, 2005, Ann Arbor.

Chen, Y., *Chinese Language Processing*, Shanghai education publishing company, 1997.

Cover, T. M., and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons Inc., New York, 1991.

Gao, J. F., H. Yu, and W. Yuan, "Minimum Sample Risk Methods for Language Modeling," In *Proceedings of Human Language Technology Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, Oct 6-8, 2005, Vancouver, Canada.

Good, I. J., "The population frequencies of species and the estimation of population parameters," *Biometrika*, 40(16), 1953, pp. 237-264.

Jeffreys, H., *Theory of Probability*, 2nd Edition, The Clarendon Press, Oxford, 1948.

Jelinek, F., *Statistical methods for speech recognition*, The MIT Press, Cambridge, Mass, 1997.

Jelinek, F., and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," In *Proceedings of the Workshop on Pattern Recognition in Practice*, Amsterdam, 1980, pp. 381-397.

Katz, S. M., "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speeech and Signal Processing*, 35(3), 1987, pp. 400-401.

Kneser, R., and H. Ney, "Improved backing-off for m-gram language modeling," In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol 1, 1995, pp. 181-184.

Kolak, O., W. Byrne, and P. Resnik, "A generative probabilistic OCR model for NLP applications," In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, Edmonton, Alberta, Canada, May 2003.

Kuhn, R., "Speech Recognition and the Frequency of Recently Used Words: A Modified Markov Model for Natural Language," In *Proceedings of 12th International Conference on Computational Linguistics (COLING 1988)*, pp. 348-350, Budapest, August 1988.

Kuhn, R., and R. D. Mori, "A Cache-Based Natural Language Model for Speech Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6), 1990, pp. 570-583.

Lafferty, J., A. McCallum, and F. Pereira, "Conditional random field: Probabilistic models for segmenting and labeling sequence data," In *Proceedings of the International Conference on Machine Learning (ICML 2001)*, 2001, pp. 282-289.

Manning, C. D., and H. Schutze, *Foundation of Statistic Natural Language Processing*, The MIT Press, 1999.

McCallum, A., D. Freitag, and F. Pereira, "Maximum Entropy Markov Models for Information Extraction and Segmentation," In *Proceedings of the International Conference on Machine Learning (ICML 2000)*, Stanford, CA, USA, 2000, pp. 591-598.

Ney, H., U. Essen, and R. Kneser, "On structuring probabilistic dependences in stochastic language modeling," *Computer, Speech, and Language*, 8, 1994, pp. 1-38.

Novak, E., and K. Ritter, "The curse of dimension and a universal method for numerical integration," In *Multivariate Approximation and Splines*, G. Nurnberger, J.W. Schmidt, G. Walz (eds.), 1998.

Rosenfeld, R, "Adaptive statistical language modeling: a maximum entropy approach," The Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA. 1994.

Xiao, J. H., B. Q. Liu, and X. L. Wang, "Principles of Non-stationary Hidden Markov Model and its Applications on Sequence Labeling Task," In *Proceedings of 2th International Joint Conference on Natural Language Processing (IJCNLP 2005)*, Lecture Notes on Artificial Intelligent, Jeju, Korea, Oct 11-13, 2005, pp. 827-837.

Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 155-174

Hierarchical Web Catalog Integration with Conceptual Relationships in a Thesaurus

Ing-Xiang Chen*, Jui-Chi Ho*, and Cheng-Zen Yang*

Abstract

Web catalog integration has become an integral aspect of current digital content management for Internet and e-commerce environments. The Web catalog integration problem concerns integration of documents in a source catalog into a destination catalog. Many investigations have focused on flattened (one-dimensional) catalogs, but few works address hierarchical Web catalog integration. This study presents a hierarchical catalog integration (EHCI) approach based on the conceptual thesauri extracted from the source catalog and the destination catalog to improve performance. Experiments involving real-world catalog integration are performed to measure the performance of the improved hierarchical catalog integration scheme. Experimental results demonstrate that the EHCI approach consistently improves the average accuracy performance of each hierarchical category.

Keywords: Hierarchical catalog integration, conceptual relationships, thesaurus, Support Vector Machines (SVMs)

1. Introduction

Automatically integrating various information sources is pertinent for many real applications given the large, and still rapidly growing, amount of information available. For instance, an on-line service provider may merge various catalogs from other on-line vendors into its local catalog to provide customers with versatile content, and a Web portal may also have to integrate different Web catalogs from other portals to provide increasingly abundant information services to users [Agrawal and Srikant 2001]. In these examples, users can gain more relevant and organized information in an integrated catalog. They can also save considerable time, because they do not need to browse different Web catalogs. According to

* Dept. of Computer Sci. and Eng., Yuan Ze University, 135 Yuan-Tung Rd., Chungli, 320, Taiwan.
Tel.: +886-3-4638800 ext: 2361 Fax: +886-3-4638850.
The author for correspondence is Cheng-Zen Yang.
E-mail: czyang@syslab.cse.yzu.edu.tw

[Received March 18, 2007; Revised June 23, 2007; Accepted July 2, 2007]

previous studies [Keller 1997; Stonebraker and Hellerstein 2001; Kim *et al.* 2002; Marrón *et al.* 2003], Web catalog integration has attracted much research interest.

Web catalog integration is not just a straightforward classification task [Agrawal and Srikant 2001]. Exploring implicit source information can effectively improve the integration accuracy [Agrawal and Srikant 2001]. Many methods for enhancing catalog integration performance have been proposed so far. The most important approach, called ENB, enhances the Naive Bayes classifiers with implicit source information. Other state-of-the-art approaches, including Support Vector Machines (SVMs) [Sarawagi *et al.* 2003; Tsay *et al.* 2003; Zhang and Lee 2004a; Chen *et al.* 2005; Chen *et al.* 2006; Ho *et al.* 2006] and the Maximum Entropy model [Wu *et al.* 2005], have been also presented to elevate the performance of Web catalog integration, and they further outperform the ENB approach.

Past studies in text classification [MacCallum *et al.* 1998; Dumais and Chen 2000] have indicated that exploiting a hierarchical structure can bring strong advantages over using a flattened structure in classification. [MacCallum *et al.* 1998] presented a probabilistic framework, and a *shrinkage* approach was proposed to improve text classification in a hierarchy of classes. Experimental results indicate that hierarchical text classification with large numbers of features (feature set > 10000) can obtain better average accuracy performance than flattened text classification. However, the shrinkage approach may either have no effect or hurt slightly in some classes with a large amount of training data [MacCallum *et al.* 1998].

Previous hierarchical data integration studies [Doan *et al.* 2002; Rajan *et al.* 2005] examined the hierarchical structures of the destination catalog are studied to improve the accuracy of catalog integration. [Doan *et al.* 2002] extracted the domain constraint features obtained from the neighboring nodes to enhance the mapping of ontological data. [Rajan *et al.* 2005] developed a maximum likelihood-based framework that exploits the hierarchical structure of categories, and examined four mapping scenarios. Experimental results have demonstrated that hierarchical relationships in the destination catalog are effective in catalog integration. Some source class labels can further be integrated into the destination catalog as new classes to maintain a new hierarchy.

However, hierarchical relationships of the categories and subcategories between the source and destination catalogs have not been investigated in the previous work. Moreover, experimental results indicate that the previously proposed approaches only integrate the data into the leaf nodes of the destination catalog. Although past methods for conventional text classification and hierarchical catalog integration can benefit from using a hierarchical structure, they only address the hierarchical structure in the destination categories and do not consider the differing hierarchical structures in the source and destination catalogs. Hence, this work performs some pilot studies for the hierarchical catalog integration problem by

considering the implicit information embedded in the hierarchical structure of both the source and destination catalogs. The pilot experimental results reported in Chen *et al.* [2006] indicate that the implicit hierarchical information does indeed contribute to the hierarchical Web catalog integration problem.

While extending the results of our previous pilot study, this work presents an enhanced hierarchical catalog integration (EHCI) approach with conceptual relationships extracted from the source and destination catalog thesauri to improve the integration performance. An EHCI approach based on SVM was adopted in these experiments due to its good classification performance. To demonstrate the effectiveness of EHCI, its performance is compared with that of a simple hierarchical catalog integration approach (SHCI) based on previous hierarchical classification studies [Dumais and Chen 2000; Sun and Lim 2001; Sun *et al.* 2003; Vural and Dy 2004].

Results of experiments with real-world catalogs reveal that the EHCI approach consistently raises the accuracy of hierarchical Web catalog integration in almost all hierarchical levels in both Yahoo!-to-Google and Google-to-Yahoo! catalog integration. These results also demonstrate that EHCI attains an average accuracy improvement of 11.1% in Yahoo!-to-Google catalog integration, and 21.6% in Google-to-Yahoo! catalog integration. The results further indicate that hierarchical catalog integration can be effectively improved by enhancing the conceptual relationships discovered from the hierarchical thesauri.

The remainder of this paper is organized as follows. Section 2 reviews the related studies of catalog integration. Section 3 then describes in detail the hierarchical Web catalog integration and the enhanced hierarchical integration approach. Next, Section 4 shows the environmental settings and discusses the experimental results. Finally, conclusions are drawn in Section 5, along with recommendations for future research.

2. Related Work

Most methods proposed for solving the catalog integration problem have been based on a flattened structure, implying that the categories in a catalog are isolated and lack hierarchical relationships. Agrawal and Srikant were the first to study this problem in 2001, and presented an enhanced Naive Bayes approach (ENB) to improve the integration accuracy by exploiting implicit information from the source catalog [Agrawal and Srikant 2001]. Experimental results involving real-world catalogs indicate that ENB can achieve an average accuracy improvement of more than 14%. Their promising results reveal that exploiting implicit source information indeed benefits the accuracy for automated catalog integration.

Several algorithms have been proposed in the past few years to increase the accuracy of catalog integration based on a flattened structure. Since SVM has presented superior

performance in classification problems [Dumais *et al.* 1998; Joachims 1998; Yang and Liu 1999; Rennie and Rifkin 2001], many related studies have also adopted the SVM classifiers with different strategies to extract the implicit information and improve the integration accuracy. These SVM-based integration approaches include a cross-training technique for SVM classifiers (SVM-CT) [Sarawagi *et al.* 2003], a topic restriction strategy (SVM-TR) [Tsay *et al.* 2003], a cluster shrinkage approach (CS-TSVM) [Zhang and Lee 2004a], and an iterative approach with pseudo-relevance feedback (SVM-IA) [Chen *et al.* 2005]. Most of these approaches employing the SVM classifiers were found to have higher accuracy then ENB.

In addition to the SVM-based approaches, some state-of-the-art investigations have also been presented to enhance the catalog integration accuracy with a flattened structure. Zhang and Lee proposed a co-bootstrapping approach with boosting to obtain the optimal combination of heterogeneous weak hypotheses without adjusting feature weights manually [Zhang and Lee 2004b]. Wu *et al*. first extracted the source hierarchical information and then applied the Maximum Entropy model to increase the accuracy of catalog integration in a flattened structure. Their experimental results showed that their approach is more accurate than ENB.

Most previous catalog integration studies adopted a flattened structure to simplify the catalog integration problem, thus neglecting the hierarchical relationships among the categories. Since previous studies on text classification problems have reported that a hierarchical structure can improve performance, an approach called *shrinkage* was presented to further improve the Bayesian classifiers in hierarchical text classification [MacCallum *et al.* 1998]. With the shrinkage-based approach, the parameter estimation of a node is smoothed by interpolation from the parent nodes, thus significantly reducing the number of prediction errors in hierarchical text classification.

Experimental results indicate that the accuracy performance of the method of MacCallum *et al*. [1998] can be raised by shrinking each leaf node with linear interpolation of the parent nodes in the destination hierarchy. However, the classification is based on the same hierarchy, instead of considering both the source and the destination hierarchies, respectively. Therefore, the original algorithm may need to be modified for application to hierarchal Web catalog integration.

Rajan *et al*. [2005] presented a two-stage mapping and integration approach, and discussed four integration scenarios. They comprehensively investigated their hierarchical catalog integration scheme using a maximum likelihood approach, and found that its integration performance is very promising, particularly in one-to-many mapping (Scenario 3). Rajan *et al*. further demonstrated that the hierarchical structure of the destination catalog is helpful in improving integration accuracy in different data sets. However, the implicit

information in the source hierarchy has not been utilized in this work.

The hierarchical relationships between the source catalog and the destination catalog requires further investigation when considering hierarchical Web catalog integration. Chen *et al*. preliminarily explored the effectiveness of a hierarchical catalog integration scheme with the consideration of both the source catalog and the destination catalog [Chen *et al.* 2006]. Their experimental results indicated a consistent improvement in accuracy of real-world Web catalog integration over the EHCI approach. Although the performance improvements are significant, the integration effectiveness based on a hierarchical structure has not been comprehensively studied. The following sections first define the problem, and then describe the ECHI approach in detail.

3. Hierarchical Web Catalog Integration

The integration process of the hierarchical catalog integration problem involves two hierarchical catalogs. Figure 1 illustrates the integration process in which the source catalog S with a set of m categories $S_1, S_2,\ldots, S_m$, is integrated into the destination catalog D with a set of n categories $D_1, D_2,\ldots, D_n$. These categories may have subcategories, such as S_{11}, D_{11} and D_{121}.

The integration process in Figure 1 is performed by merging each document d_i in S into a correspondent destination category in D. Thus, for each directory in the hierarchy, the training documents trained as directory classifiers and local classifiers are utilized to help integrate each document d_i into a corresponding directory. Only the documents integrated into the corresponding level categories and subcategories are regarded as correctly integrated.

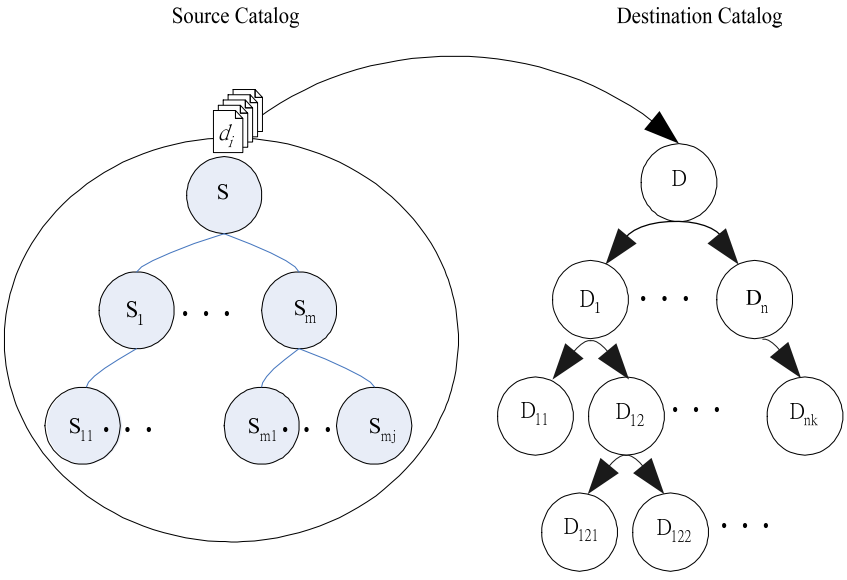


Figure 1. The process of hierarchical catalog integration.

This study adopts SVM classifiers with linear kernel functions [Yang and Liu 1999], $f : X \in R^n \rightarrow R$ to locate a hyperplane that can separate the positive examples, $f(x) \geq +1$, from the negative examples, $f(x) \leq -1$. The linear function is in the form $f(x) = (x,b) + b = \sum_{i=1}^{n} wixi + b$ where $(w,b) \in R^n \rightarrow R$. The linear SVM is trained to determine the optimal values of *w* and *b* such that ||*w*|| is minimized. These trained SVM classifiers are employed in the simple hierarchical catalog integration (SHCI) approach and the enhanced hierarchical catalog integration (EHCI) approach in hierarchical catalog integration. The SHCI approach and the EHCI scheme are described as follows.

3.1 The Simple Hierarchical Catalog Integration (SHCI) Approach

In SHCI, the SVM classifiers are trained with the training documents coming from the destination catalog and are used to integrate the test documents from the source catalog into the destination catalog. Whether a training document is considered a positive document or a negative document depends on its subordinate relationship to each destination category. Referring to Sun and Lim [2001] and Sun *et al.* [2003], the destination catalog was designed with two classifiers at every category node, namely a directory classifier and a local classifier.

The directory classifiers were designed to categorize the source documents into different category and subcategory trees. The directory classifiers are trained with equal numbers of positive and negative examples. The positive examples were chosen from the categories and their subcategories where the documents were located. The negative examples were selected from the remaining categories and their subcategories under the same level. The local classifiers were designed to classify the source documents further into different destination levels in each category tree. The local classifiers in each level were trained with the positive examples chosen from each destination level, and the negative examples selected from the subcategories under that level.

In real-world Web catalogs, a document may be integrated into more than one category. Therefore, a "one-against-rest" strategy was adopted to extend the binary SVM classifiers and solve the multi-class catalog integration problem. This study uses the SHCI approach as a baseline for hierarchical catalog integration, and considers the performance improvement of the SVM classifiers resulting from the enhancement of conceptual relationships in thesauri.

3.2 Conceptual Relationships in Web Thesaurus

Foskett utilized a thesaurus as a dictionary and a reference for classification [Foskett 1997]. A thesaurus can be defined as a set of related terms in a given domain knowledge, and these related terms are the basic semantic units for conveying concepts [Wikipedia: thesaurus]. Since a hierarchical thesaurus defines broad and narrow terms, its classification system can be considered a vocabulary hierarchy. Likewise, the child nodes in a hierarchical Web catalog

structure generally comprise related terms to express the classified concepts of the parent nodes, and so the classified terms in a hierarchical Web catalog can be treated as a hierarchical thesaurus. Figure 2 shows an example in which the "Automotive" category in Yahoo! Web catalog is categorized like a hierarchical thesaurus with some conceptual relationships in the hierarchy.

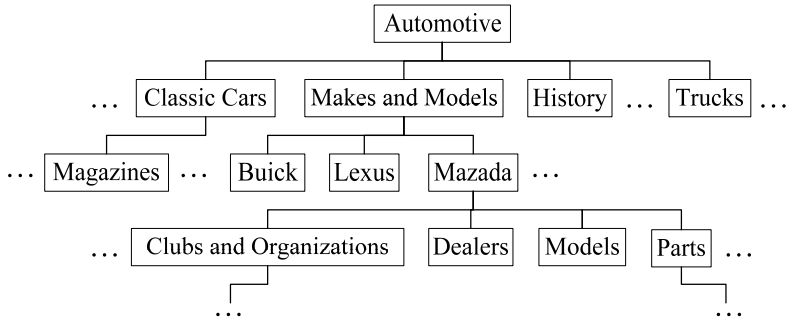


Figure 2. The illustration of a Web thesaurus in Yahoo! catalog

In Figure 2, the term "Automotive" is the thesaurus root, which expresses a broad term in the hierarchy, and has different narrow terms to define different types of "Automotive". Narrower terms are defined down to the leaf nodes in the hierarchy. In the Web catalog hierarchy, the conceptual relationships can be extracted from the hierarchical thesaurus and can construct different semantic concepts. Therefore, different domain knowledge can be extracted from the Web catalog hierarchy, thus enhancing the performance of the SVM classifiers.

3.3 Enhanced Hierarchical Catalog Integration (EHCI) Scheme

To elevate the integration performance, a weighting formula, Equation (1), is designed to exploit the conceptual relationships from the hierarchical Web thesaurus, where the terms in different category levels are extracted as label features. Equation (1) calculates the feature weight of each document, $FeatureWeight_{(x,\ d)}$, where L_i denotes the relevant label weight assigned exponentially as $1/2^i$, f_x represents the occurrence ratio of feature x in the document, and λ indicates the magnitude relation of the label weight. In Equation (1), the weight of each thesaurus is exponentially decreased and accumulated based on the increased levels, where n denotes the depth of a document in the hierarchy. If feature x appears in the label feature, then L_x is denoted as the label weight with the level where x is located. Otherwise, L_x=0. Consequently, Equation (1) is applied to both the source and destination hierarchies to represent the semantic concepts obtained from the source category labels and the destination category labels.

$$FeatureWeight_{(x,\ d)} = \lambda \times \frac{L_x}{\sum_{i=0}^{n} L_i} + (1-\lambda) \times f_x \quad (1)$$

Table 1. The label weights assigned for different hierarchical levels

Hierarchical Level	Label Weight
Document Level (L_0)	$1/2^0$
One Level Upper (L_1)	$1/2^1$
Two Levels Upper (L_2)	$1/2^2$
…	…
n Levels Upper (L_n)	$1/2^n$

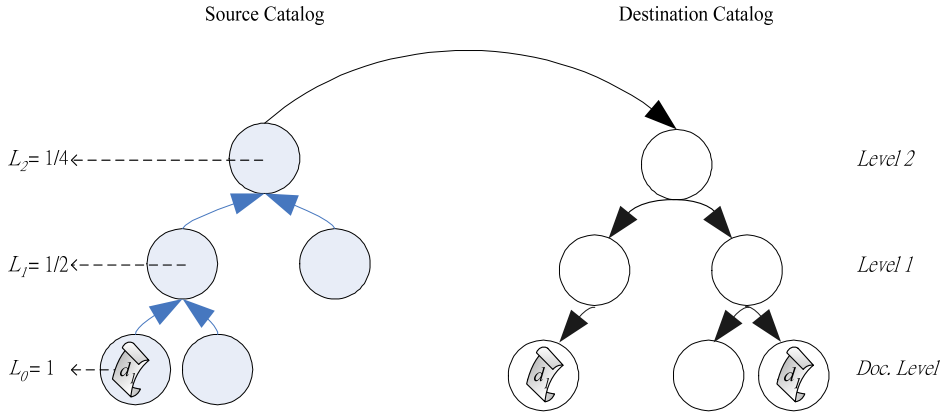


Figure 3. The process of the enhanced hierarchical catalog integration

In Equation (1), L_i further denotes the label weight at a depth of i. The label weight falls from the document level (i=0) to top level n. This thesaurus weighting method can be utilized to transform the conceptual relationships of the hierarchical source categories, and add them into the test documents. Table 1 lists the weights of different hierarchical labels, where L_0 denotes the document level; L_1 represents one level above, and so on down to L_n representing n levels above.

Similarly, the EHCI scheme is used in the destination catalog to build enhanced classifiers in destination categories. With the enhancement of the features and native category label information, the classifiers can thus be trained to be more distinctive to classify the documents into the correct categories. The weights of the features and native category label information in the destination catalog are also calculated according to Equation (1). The threshold λ is set with different values from 0 to 1 to find the optimized weights for the source thesauri to enhance the destination classifiers, as are the values of λ set in the native destination category. Moreover, the features occurring in the upper categories are removed to avoid misleading integration in the subcategories.

Figure 3 displays a three-level example to demonstrate the concept of the EHCI approach. In the source catalog, the hierarchical thesaurus information is added to the test documents with different label weights accumulated upward from their current categories to the top-level category according to the weighting formula. In the destination catalog, the test documents are integrated into the destination categories based on the EHCI integration scheme. Figure 3 also indicates that a document d_1 may be integrated into more than one destination category.

3.4 Enhanced Catalog Integration Process

Since a Web document generally comprises HTML tags, script codes and texts, the HTML tags and scripts codes are eliminated, and only the texts obtained after retrieving the Web documents from both the source and destination catalogs are kept. In the preprocessing stage, the texts are segmented into terms by removing the stopwords and stemming the terms with the Porter Stemmer [Porter 1980]. The weight of each stemmed term x is assigned by $TFx/\sum TFi$, where i denotes the number of the stemmed terms in each document. This preprocessing flow and feature weight strategy is applied to both the SHCI and EHCI schemes.

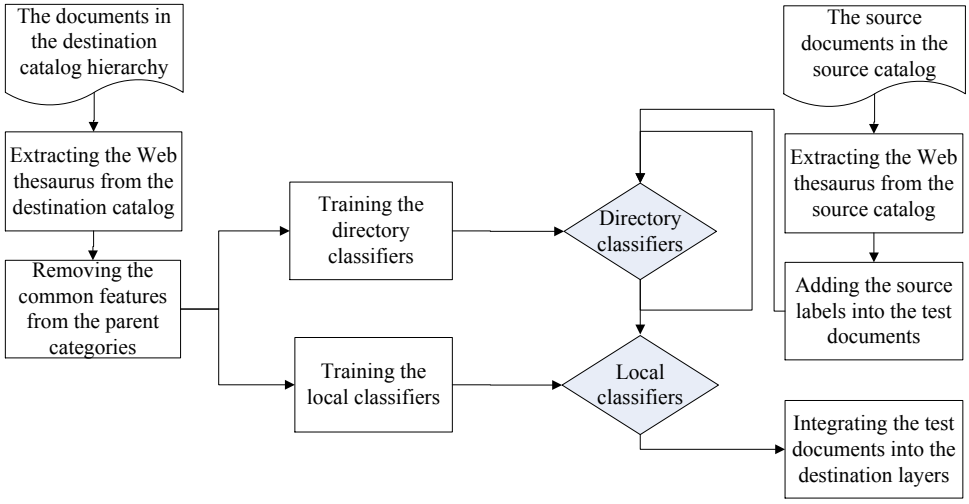


Figure 4. The process of enhanced hierarchical catalog integration

Figure 4 shows the process of hierarchical catalog integration with the EHCI scheme. In the integration process, the terms transformed from the test documents are added with the source catalog labels based on Equation (1). Similarly, the terms transformed from the documents in the destination categories are trained using the labels extracted from the destination catalog. To establish the directory classifiers and local classifiers in the destination catalog, the common features in the parent categories are removed in the training stage to avoid building ambiguous classifiers.

In Figure 4, the directory classifiers are trained with the positive documents from their categories and subcategories to represent the classifiers of the category trees. The local classifiers are trained by the positive documents of the same levels to represent the classifiers of their local levels. The selection of negative examples in the directory classifiers and the local classifiers is similar to the SHCI approach as described in Section 3.1. The test documents are then integrated into the destination categories through both the directory classifiers and the local classifiers. The integration process is finished when all the test documents from their source categories are integrated into the designated destination categories.

4. Experiments and Discussion

Experiments were performed involving real-world catalogs from both Yahoo! and Google to examine the performance of the EHCI schemes with SVM ^{light} [Joachims 2002]. The average integration performance with different λ values between 0 and 1 were compared. The results with the optimal λ value are listed in detail. Experimental results indicate that the EHCI approach consistently enhances the SVM classifiers in almost all levels and boosts the integration accuracy of a hierarchical structure. The following subsections describe the data sets and the experimental results.

4.1 Data Sets

Five categories were extracted from Yahoo! and Google. Table 2 shows the statistics of our experimental data including the number of hierarchical classes, the training documents and the test documents in these five categories. The experimental data were collected after neglecting the documents that could not be retrieved and removing the documents with error messages. The stopword list in Frakes and Baeza-Yates [1992] was adopted to remove the stopwords in preprocessing. Over 38,000 terms were employed for training and testing after removing the stopwords and stemming. As in [Agrawal and Srikant 2001], documents appearing in only one catalog were used as the training documents in the destination catalog D, and the common documents were adopted as the test documents in the source catalog S.

Table 2. The experimental data collected from the Google catalog

Category	Google	|G-Y|	|G Class|	|G Test|	Yahoo!	|Y-G|	|Y Class|	|Y Test|
Autos	…/Autos/…	1094	312	437	…/Automotive/…	1823	148	404
Movies	…/Movies/…	5174	1165	1340	…/Movies_Film/…	7776	1035	1211
Outdoors	…/Outdoors/…	2308	523	224	…/Outdoors/…	1724	100	177
Photo	…/Photography/…	615	158	206	…/Photography/…	1399	80	175
Software	…/Software/…	5693	1185	683	…/Software/…	1940	109	646
Total		14884	3343	2890		14662	1472	2613

A set of 1,472 classes in the Yahoo! catalog and a set of 3,343 classes in the Google catalog were organized according to the original hierarchy to a depth of six levels as shown in Table 2. The test documents were chosen by cross-referencing the documents of Yahoo! with those of Google. Table 2 indicates that the numbers of test documents in Yahoo! and Google were different, in the sense that some test documents may appear in more than one class simultaneously. The training documents of the Yahoo! catalog and the Google catalog were accumulated by subtracting the common documents in the other catalog. In this experiment, the documents were integrated both from Yahoo! into Google and from Google to Yahoo!.

Tables 3 and 4 further describe the number of the hierarchical classes, the training documents, and the test documents of six levels in the Google and Yahoo! catalogs. Since most of the sixth levels contain less than ten documents, the hierarchies were only retrieved down to the sixth level, and any documents below the sixth level were merged upward to the sixth level. Tables 3 and 4 indicate that the numbers of some Level 1 classes were zero, meaning that the destination category contained no Level 1 test documents. This experiment only considered the documents that were correctly integrated into the destination categories, thus we list the number of classes with common test documents.

Table 3. The experimental data collected from the Google catalog

	Level 1	Level 2	Level3	Level 4	Level 5	Level 6	Total
Class # in Autos	0	14	98	148	46	6	312
Training doc.# in Autos	0	144	422	389	127	12	1094
Test doc. # in Autos	0	86	218	111	19	3	437
Class # in Movies	1	27	115	700	245	77	1165
Training doc.# in Movies	3	136	2581	1554	718	182	5174
Test doc. # in Movies	0	131	524	348	302	35	1340
Class # in Outdoors	1	23	114	111	104	170	523
Training doc.# in Outdoors	1	104	594	376	434	799	2308
Test doc. # in Outdoors	0	40	76	69	24	15	224
Class # in Photo	0	9	29	50	52	18	158
Training doc.# in Photo	0	28	172	227	141	47	615
Test doc. # in Photo	0	26	88	59	25	8	206
Class # in Software	1	59	281	352	306	186	1185
Training doc.# in Software	2	547	1784	1656	1189	515	5693
Test doc. # in Software	2	29	149	241	157	105	683

Table 4. The experimental data collected from the Yahoo! catalog

	Level 1	Level 2	Level3	Level 4	Level 5	Level 6	Total
Class # in Autos	1	24	61	39	17	6	148
Training doc.# in Autos	56	490	575	467	186	49	1823
Test doc. # in Autos	11	126	119	101	38	9	404
Class # in Movies	1	27	91	195	584	137	1035
Training doc.# in Movies	2	653	992	2210	3260	659	7776
Test doc. # in Movies	0	140	180	353	404	134	1211
Class # in Outdoors	1	26	47	17	5	4	100
Training doc.# in Outdoors	63	455	815	305	25	61	1724
Test doc. # in Outdoors	0	44	114	18	0	1	177
Class # in Photo	1	18	28	14	17	2	80
Training doc.# in Photo	28	266	453	138	496	18	1399
Test doc. # in Photo	1	72	78	19	5	0	175
Class # in Software	1	15	24	24	26	19	109
Training doc.# in Software	50	366	488	489	364	183	1940
Test doc. # in Software	3	146	133	155	174	35	646

Table 5. The analysis of common classes between Google and Yahoo!

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
Common class # in Autos	-	2	39	1	0	0
Common class # in Movies	-	16	24	9	6	1
Common class # in Outdoors	-	5	3	2	3	0
Common class # in Photo	-	4	2	1	3	0
Common class # in Software	-	7	6	8	8	3

Since hierarchical catalog integration is not like hierarchical text classification on the basis of the same hierarchy, the structure of the source hierarchy can be very different from the structure of the destination hierarchy. Table 5 further analyzes the number of common classes in different levels between the Yahoo! catalog and the Google catalog. Table 5 indicates that the number of common classes from Level 2 to Level 6 was very small. For example, the Level 2 category of "Autos" contains only two common classes (chats_and_forums and makes_and_models) between the Google catalog (14 classes) and the Yahoo! catalog (24 classes).

In addition to the common classes in the same Level 1 categories, the common classes in different Level 1 categories were also analyzed. The results reveal that the different Level 1 categories had very few common classes or even no common classes in other hierarchical

subcategories. For instance, Yahoo! "Outdoor" has only one common Level 2 subcategory in Google "Movie", and no common Level 2 subcategories in Google "Autos", "Photo", and "Software". Prior analysis reveals that the hierarchical structure of the source catalog in the real-world experimental data is different from that of the destination catalog.

4.2 Measurement

Since some documents may appear in more than one category of the same catalog, the number of test documents may vary slightly between Yahoo! and Google. This experiment followed an assumption in Agrawal and Srikant [2001] by measuring the performance of hierarchical catalog integration with accuracy defined in the following equation.

$$\frac{\text{Number of the test documents correctly integrated into } Di}{\text{Total number of the test documents in the dataset}} \quad (2)$$

To measure the performance of hierarchical catalog integration, Equation (2) was adopted in each level of the destination categories to assess its accuracy performance. In each level of the destination categories, the numerator denotes the test documents correctly integrated into that level, and the denominator represents the total test documents to be correctly integrated. The accuracy of each level in the destination categories and the average accuracy of the five categories were measured.

4.3 Results and Discussion

In the experiment, the documents were integrated both from Yahoo! into Google and from Google to Yahoo!. In the EHCI approach, the conceptual relationships between the hierarchical thesauri in both the source and destination categories added to an increasing λ value in the range 0–1. To further verify the effectiveness of the EHCI approach, three sets of negative examples were randomly chosen for Google training and the other three sets of negative examples were used for Yahoo! training. The overall performance of the EHCI approach is significantly boosted in all of these six sets. The best average performance improvements from Yahoo! to Google with the three sets of negative examples were 9.0%, 11.1%, and 21.6%. In contrast, the best performance improvements from Google to Yahoo! with the other three sets of negative examples were 18.1%, 21.6%, and 23.7%. Table 6 and Table 7, notably, list the medians and detail the average integration results with λ increasing from 0 to 1.

Table 6 shows the average catalog integration performance from Yahoo! to Google, and Table 7 lists that from Google to Yahoo!. Both first columns represent the λ values of the source catalog, and the first rows represent the λ values of the destination catalog. As indicated in Tables 6 and 7, the accuracy with the SHCI approach ($\lambda = 0.00$) from Yahoo! to

Google was 61.4% and that from Google to Yahoo! was 63.7%. The best performance improvements with EHCI were achieved at $\lambda = 0.01$ in the destination catalog and $\lambda = 0.30$ in the source catalog. The average accuracy from Yahoo! to Google and Google to Yahoo! was 72.5% and 85.3%, respectively.

Table 6. The average integration performance from Yahoo! to Google

S \ D	0.00	0.01	0.05	0.10	0.30	0.50	0.70	0.90	1.00
0.00	61.4%	61.1%	52.4%	38.4%	17.8%	15.1%	14.2%	13.9%	13.7%
0.01	60.7%	62.3%	54.9%	40.7%	18.4%	15.2%	14.3%	14.0%	13.7%
0.05	63.6%	66.2%	63.4%	52.1%	21.2%	15.9%	14.6%	14.2%	14.0%
0.10	66.3%	69.6%	68.0%	60.3%	27.6%	17.6%	15.3%	14.3%	14.2%
0.30	68.7%	**72.5%**	72.1%	68.5%	54.7%	35.9%	26.2%	18.2%	17.4%
0.50	67.1%	71.2%	71.7%	69.9%	61.6%	53.7%	40.1%	30.8%	28.0%
0.70	64.6%	68.5%	69.1%	68.2%	61.2%	58.7%	52.2%	41.3%	37.9%
0.90	64.1%	67.8%	69.0%	68.5%	62.7%	60.5%	56.8%	51.9%	47.6%
1.00	63.6%	67.4%	68.9%	68.5%	63.3%	61.5%	58.5%	53.8%	51.8%

Table 7. The average integration performance from to Google to Yahoo!

S \ D	0.00	0.01	0.05	0.10	0.30	0.50	0.70	0.90	1.00
0.00	63.7%	61.7%	33.1%	15.0%	0.8%	0.2%	0.1%	0.1%	0.1%
0.01	66.0%	65.6%	37.8%	16.9%	1.0%	0.2%	0.1%	0.2%	0.1%
0.05	72.4%	74.2%	54.2%	29.4%	1.8%	0.3%	0.2%	0.2%	0.1%
0.10	76.7%	80.4%	64.9%	42.2%	4.6%	0.6%	0.2%	0.2%	0.2%
0.30	81.8%	**85.3%**	75.5%	57.1%	29.7%	12.2%	2.3%	0.3%	0.2%
0.50	80.2%	85.0%	77.7%	60.4%	39.6%	26.2%	16.4%	8.4%	4.3%
0.70	77.5%	83.5%	77.8%	61.7%	43.2%	32.7%	24.3%	19.6%	14.5%
0.90	75.8%	82.5%	78.1%	62.9%	45.9%	38.5%	30.0%	24.2%	21.0%
1.00	75.2%	81.7%	78.3%	63.1%	46.6%	38.9%	32.0%	26.2%	24.2%

Since the best accuracy in both Google-to-Yahoo! and Yahoo!-to-Google integration was obtained by adding the hierarchical label weights with $\lambda = 0.30$, we can infer that the conceptual thesaurus extracted from the source hierarchy significantly improves hierarchical catalog integration. Conversely, the conceptual thesaurus extracted from the destination hierarchy to enhance the hierarchical classifiers is not as effective as the source hierarchical thesaurus. Tables 6 and 7 show that the improvement in accuracy obtained by changing from $\lambda = 0.00$ to $\lambda = 0.01$ was less than 5%. Experimental results indicate that the conceptual relationships in the source hierarchical thesaurus are more likely to enhance hierarchal Web catalog integration than those in the destination hierarchical thesaurus.

Tables 8 and 9 further describe the integration accuracy of the six hierarchical levels with $\lambda = 0.01$ in the destination catalog and $\lambda = 0.30$ in the source catalog. Analytical results indicate that the EHCI approach consistently improves the accuracy performance of each level in almost all cases. However, Table 8 still indicates that the EHCI approach induced a 2.7% accuracy decrease at Level 3 and a 23.8% accuracy decrease at Level 6 in the Software category when integrating from Yahoo! to Google. Table 9 also indicates a 0.7% accuracy decrease at Level 6 in the Movies category, and a 5.7% accuracy decrease at Level 6 in the Software category when integrating from Google to Yahoo!. The main reason for these falls in accuracy is probably due to the training documents in those destination levels lacking the hierarchical thesauri extracted from the source catalog.

Table 8. The Yahoo!-to-Google integration performance in six levels

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Total
Autos	0.0% (0)	84.9% (73)	50.0% (109)	40.5% (45)	52.6% (10)	0.0% (0)	54.2% (237)
Autos_E	0.0% (0)	91.9% (79)	74.8% (163)	60.4% (67)	68.4% (13)	66.7% (2)	74.1% (324)
Movies	0.0% (0)	63.4% (83)	75.4% (395)	56.6% (197)	54.0% (163)	37.1% (13)	63.5% (851)
Movies_E	0.0% (0)	71.8% (94)	80.0% (419)	61.5% (214)	70.2% (212)	40.0% (14)	71.1% (953)
Outdoors	0.0% (0)	70.0% (28)	75.0% (57)	69.6% (48)	79.2% (19)	46.7% (7)	71.0% (159)
Outdoors_E	0.0% (0)	72.5% (29)	93.4% (71)	87.0% (60)	87.5% (21)	73.3% (11)	85.7% (192)
Photo	0.0% (0)	42.3% (11)	55.7% (49)	39.0% (23)	36.0% (9)	25.0% (2)	45.6% (94)
Photo_E	0.0% (0)	61.5% (16)	68.2% (60)	57.6% (34)	52.0% (13)	25.0% (2)	60.7% (125)
Software	100.0% (2)	72.4% (21)	63.1% (94)	68.0% (164)	58.0% (91)	58.1% (61)	63.4% (433)
Software_E	100.0% (2)	86.2% (25)	60.4% (90)	82.6% (199)	73.2% (115)	34.3% (36)	68.4% (467)

Table 9. The Google-to-Yahoo! integration performance in six levels

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Total
Autos	63.6% (7)	68.3% (86)	64.7% (77)	52.5% (53)	47.4% (18)	88.9% (8)	61.6% (249)
Autos_E	100.0% (11)	84.9% (107)	89.1% (106)	88.1% (89)	84.2% (32)	100.0% (9)	87.6% (354)
Movies	0.0% (0)	72.1% (101)	56.7% (102)	50.1% (177)	53.7% (217)	50.7% (68)	54.9% (665)
Movies_E	0.0% (0)	91.4% (128)	78.9% (142)	94.9% (335)	65.6% (265)	50.0% (67)	77.4% (937)
Outdoors	0.0% (0)	70.5%(31)	80.7% (92)	44.4% (8)	0.0% (0)	100.0% (1)	74.6% (132)
Outdoors_E	0.0% (0)	100.0% (44)	97.4% (111)	88.9% (16)	0.0% (0)	100.0% (1)	97.2% (172)
Photo	0.0% (0)	63.9% (46)	60.3% (47)	84.2% (16)	40.0% (2)	0.0% (0)	63.4% (111)
Photo_E	0.0% (0)	90.3% (65)	93.6% (73)	84.2% (16)	60.0% (3)	0.0% (0)	89.7% (157)
Software	100.0% (3)	83.6% (122)	75.9% (101)	76.1% (118)	79.9% (139)	71.4% (25)	78.6% (508)
Software_E	100.0% (3)	93.8% (137)	85.0% (113)	91.6% (142)	97.1% (169)	65.7% (23)	90.9% (587)

Figures 5 and 6 depict the overall performance between the EHCI and SHCI approaches. The results indicate that EHCI outperforms SHCI in both Yahoo!-to-Google and Google-to-Yahoo! catalog integration. Figure 5 indicates that the EHCI approach achieved an average accuracy improvement of 11.1% in Yahoo!-to-Google catalog integration. In Figure 6, the EHCI approach obtained an average accuracy improvement of 21.6% in Google-to-Yahoo! catalog integration. The results further indicate that hierarchical catalog integration can be effectively boosted by enhancement of the conceptual relationships extracted from the hierarchical thesauri.

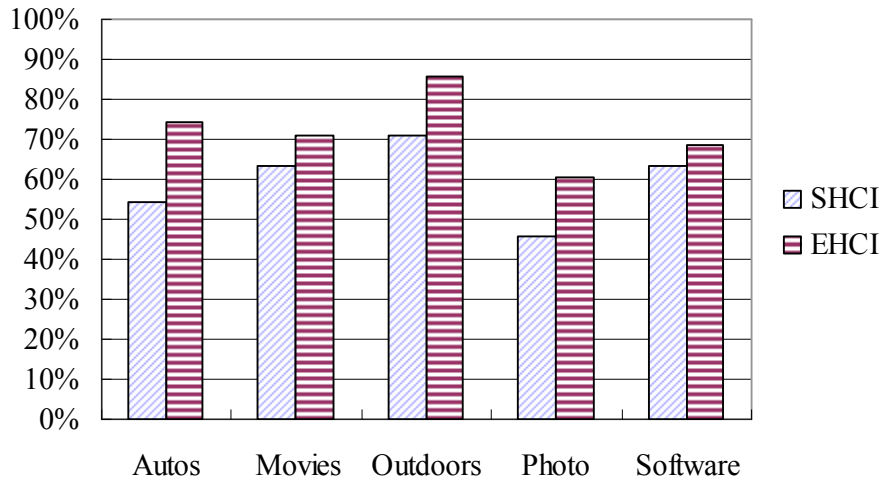


Figure 5. The average integration performance from Yahoo! to Google

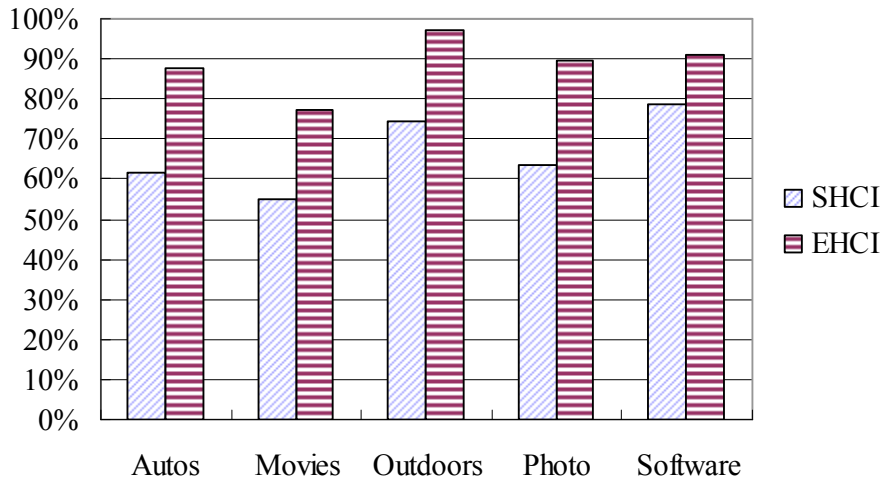


Figure 6. The average integration performance from Google to Yahoo!

As well as the accuracy performance, the computation cost of SHCI and EHCI approaches was further analyzed. The experimental environment was in an IBM PC with an Intel Core Duo T2400 CPU and 1GB memory. The overall CPU runtime provided by SVM^{light} was to analyze the training and testing time, excluding the data I/O time, in a Windows XP environment. Results of runtime analysis demonstrate that SHCI took 65.60 seconds to perform Google-to-Yahoo! catalog integration and 6.40 seconds to perform Yahoo!-to-Google catalog integration. Conversely, EHCI took 86.53 seconds to perform Google-to-Yahoo! catalog integration and 6.69 seconds to perform Yahoo!-to-Google catalog integration. The reason for the faster CPU time of Yahoo!-to-Google integration is the much smaller number of Google classifiers than Yahoo! classifiers. The CPU runtime analysis further indicates that the proposed approach can efficiently complete the catalog integration work.

5. Conclusion

Web catalog integration is a significant issue in Web content management. Although past studies have indicated that a hierarchical structure is superior to a flattened structure in classification, recent studies have only presented a few primitive results and have not comprehensively studied hierarchical structures in hierarchical Web catalog integration. This study addresses the problem of hierarchical catalog integration, and proposes an enhanced hierarchical catalog integration (EHCI) scheme.

This study further reports experimental results concerning the improvement in Web catalog integration accuracy resulting from the use of EHCI. The integration accuracy is significantly improved by exploiting the conceptual relationships extracted from the source and destination catalog thesauri to enhance hierarchical catalog integration. Experimental results indicate that EHCI is effective for hierarchical Web catalog integration, and achieves improvements in almost every hierarchical level on real-world catalogs with SVM classifiers. In overall performance of hierarchical catalog integration, the EHCI approach can consistently improve accuracy in real-world catalog integration.

To conclude, this study demonstrates that the conceptual relationships learned from the source and destination catalog thesauri can enhance hierarchical catalog integration. Experimental results indicate that the accuracy improvements in a hierarchical structure are very promising, especially the hierarchical thesaurus extracted from the source catalog. Future work will involve investigating other classification models in order to build an integration platform for hierarchical catalog integration. Furthermore, complex catalog integration issues will be considered through ontology relationships.

Acknowledgement

The authors would like to thank the National Science Council of the Republic of China, Taiwan, for partially supporting this research under Contract No. NSC 95-2745-E-155-008. The authors would also like to express many thanks to the anonymous reviewers for their precious suggestions.

References

Agrawal, R., and R. Srikant., "On Integrating Catalogs," in *Proceedings of the 10th WWW Conf. (WWW10)*, Hong Kong, 2001, pp. 603–612.

Chen, I.-X., C.-Z. Yang, and J.-C. Ho, "An Iterative Approach for Web Catalog Integration with Support Vector Machines," in *Proceedings of Asia Information Retrieval Symposium 2005 (AIRS2005)*, Jeju Island, Korea, 2005, pp. 703–708.

Chen, I.-X., C.-Z. Yang, and J.-C., Ho, "On Hierarchical Web Catalog Integration with Conceptual Relationships in Thesaurus," in *Proceedings of the 29th Annual ACM Conf. on Research and Development in Information Retrieval (SIGIR'06)*, Settle, Washington, USA, 2006, pp. 635–636.

Doan, A., J. Madhavan, P. Domingos, and A. Halevy, "Learning to Map between Ontologies on the Semantic Web," in *Proceedings of the 11th WWW Conf. (WWW2002)*, Honolulu, Hawaii, 2002, pp. 662–673.

Dumais, S., J. Platt, D. Heckerman, and M. Sahami, "Inductive Learning Algorithms and Representations for Text Categorization," in *Proceedings of the 7th Int'l Conf. on Information and Knowledge Managemen (CIKM)*, Bethesda, Maryland, USA, 1998, pp. 148–155.

Dumais, S., and H. Chen, "Hierarchical Classification of Web Content," in *Proceedings of the 23rd Annual ACM Conf. on Research and Development in Information Retrieval (SIGIR'00)*, Athens, Greece, 2000, pp. 256–263.

Frakes, W., and R. Baeza-Yates, *Information Retrieval: Data Structures and Algorithms*, Prentice Hall Press, USA, 1992.

Foskett, D.J., "Thesaurus," in *Readings in Information Retrieval*, Jones, K.S., and Willett, P., Ed. Morgan Kaufmann Press, San Francisco, CA, USA, 1997, pp. 111 –134.

Ho, J.-C., I.-X. Chen, and C.-Z. Yang, "Learning to Integrate Web Catalogs with Conceptual Relationships in Hierarchical Thesaurus," in *Proceedings of Asia Information Retrieval Symposium 2006 (AIRS2006)*, Singapore, 2006, pp. 217–229.

Joachims, T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," in *Proceedings of the 10th European Conf. on Machine Learning (ECML'98)*, Chemnitz, DE, 1998, pp. 137–142.

Keller, A. M., "Smart Catalogs and Virtual Catalogs," in *Readings in Electronic Commerce*, Kalakota, R., and Whinston, A., Ed. Addison-Wesley Press, USA, 1997.

Kim, D., J. Kim, and S. Lee, "Catalog Integration for Electronic Commerce through Category-Hierarchy Merging Technique," in *Proceedings of the 12th Int'l Workshop on Research Issues in Data Engineering: Engineering e-Commerce/e-Business Systems (RIDE'02)*, San Jose, CA, USA, 2002, pp. 28–33.

MacCallum, A., R. Rosenfeld, T. Mitchell, and A. Ng, "Improving Text Classification by Shrinkage in a Hierarchy of Classes," in *Proceedings of the 15th International Conference on Machine Learning (ICML-98),* Madison, Wisconsin, 1998, pp. 359–367.

Marrón, P. J., G. Lausen, and M. Weber, "Catalog Integration Made Easy," in *Proceedings of the 19th Int'l Conf. on Data Engineering (ICDE'03)*, Bangalore, India, 2003, pp. 677–679.

Rajan, S., K. Punera, and J. Ghosh, "A Maximum Likelihood Framework for Integrating Taxonomies", *in Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05),* Pittsburgh, Pennsylvania, pp. 856–861.

Rennie, J. D. M., and R. Rifkin, "Improving Multiclass Text Classification with the Support Vector Machine," *Technical Report AI Memo AIM-2001-026 and CCL Memo 210*, MIT Press, USA, 2001.

Sarawagi, S., S. Chakrabarti, and S. Godbole, "Cross-Training: Learning Probabilistic Mappings between Topics," in *Proceedings of the 9th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, Washington, D.C., 2003, pp. 177–186.

Stonebraker, M., and J. M. Hellerstein, "Content Integration for e-Commerce," in *Proceedings of the 2001 ACM SIGMOD Int'l Conf. on Management of Data*, Santa Barbara, CA, USA, 2001, pp. 552–560.

Sun, A., and E.-P. Lim, "Hierarchical Text Classification and Evaluation," in *Proceedings of the 2001 IEEE Int'l Conf. on Data Mining (ICDM'01)*, Washington, D.C., USA, 2001, pp. 521–528.

Sun, A., E.-P. Lim, and W.-K. Ng, "Performance Measurement Framework for Hierarchical Text Classification," *Journal of the American Society for Information Science and Technology (JASIST)*, 54(11), 2003, pp. 1014–1028.

Tsay, J.-J., H.-Y. Chen, C.-F. Chang, and C.-H. Lin, "Enhancing Techniques for Efficient Topic Hierarchy Integration," in *Proceedings of the 3rd Int'l Conf. on Data Mining (ICDM'03)*, Melbourne, FL, USA, 2003, pp. 657–660.

Vural, V., and J. G. Dy, "A Hierarchical method for Multi-Class Support Vector Machine," in *Proceedings of the Int'l Conf. on Machine Learning 2004 (ICML2004)*, Banff, Alberta, Canada, 2004, pp. 105.

Wu, C.-W., T.-H. Tsai, and W.-L. Hsu, "Learning to Integrate Web Taxonomies with Fine-Grained Relations: A Case Study Using Maximum Entropy Model," in *Proceedings of Asia Information Retrieval Symposium 2005 (AIRS2005)*, Jeju Island, Korea, 2005, pp. 190–205.

Yang, Y., and X. Liu, "A Re-examination of Text Categorization Methods," in *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval (SIGIR'99)*, Berkeley, CA, USA, 1999, pp. 42–49.

Zhang, D., and W.S. Lee, "Web Taxonomy Integration using Support Vector Machines," in *Proceedings of the 13th WWW Conf. (WWW2004),* New York, NY, USA, 2004a, pp. 472–481.

Zhang, D., and W.S. Lee, "Web Taxonomy Integration through Co-Bootstrapping," in *Proceedings of the 27rd Annual ACM Conf. on Research and Development in Information Retrieval (SIGIR'04)*, Athens, Greece, Sheffield, United Kingdom, 2004b, pp. 410–417.

Online Resources

Google Catalog: http://www.google.com/dirhp

Joachims, T., SVM*ligh*:, version 5.0, http://svmlight.joachims.org/, 2002

Porter, M., Porter Stemmer: http://www.tartarus.org/martin/PorterStemmer/, 1980

Yahoo! Catalog: http://dir.yahoo.com/

Wikipedia http://en.wikipedia.org/wiki/Thesaurus

Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 175-194

MiniJudge: Software for Small-Scale Experimental Syntax

James Myers*

Abstract

MiniJudge is free online open-source software to help theoretical syntacticians collect and analyze native-speaker acceptability judgments in a way that combines the speed and ease of traditional introspective methods with the power and statistical validity afforded by rigorous experimental protocols. This paper shows why MiniJudge is useful, what it feels like to use it, and how it works.

Keywords: Syntax, Experimental Linguistics, JavaScript, R, Generalized Linear Mixed Effect Modeling

1. Introduction

Every theoretical syntactician has faced the problem of native-speaker judgments that, instead of correlating neatly with the theoretical issue at hand, vary unexpectedly across sentences or speakers. This problem is generally dealt with indiscriminately, either by fiat ("assuming these judgments are correct...") or by dropping the data entirely, along with the potentially important theoretical issue it may provide. Perhaps forty years ago [Chomsky 1965:19-20] was right to declare that "[t]he critical problem for grammatical theory today is not a paucity of evidence but rather the inadequacy of present theories of language to account for masses of evidence that are hardly open to serious question." However, as [Schütze 1996:27] observed (ten years ago now), "the questions linguists are now addressing rely crucially on facts that are indeed 'open to serious question'."

Acceptability judgments reflect grammatical knowledge, but as data they are merely a form of linguistic behavior, parallel to the accuracy rates or reaction times measured by psycholinguists ([Chomsky 1965], [Penke and Rosenbach 2004]). From a cognitive science perspective, then, the ideal solution to the linguists' data woes would be for them to adopt the rigorous experimental protocols honed over the two centuries scientists have been struggling

* Graduate Institute of Linguistics, National Chung Cheng University, 168 University Road, Min-Hsiung, Chia-Yi 62102, Taiwan Phone: 886-5-242-8251 Fax: 886-5-272-1654
E-mail: lngmyers@ccu.edu.tw

[Received November 26, 2006; Revised July 3, 2007; Accepted July 11, 2007]

to extract information about mental structure from often messy behavioral data. When linguistic judgments are collected with such protocols, they often (though not always) reconfirm the essential validity of empirical claims made on the basis of more informal methods, but they can also go beyond simple reconfirmation (or falsification) to reveal hitherto unsuspected theoretical insights. Recent examples of the growing experimental syntax literature include [Sorace and Keller 2005], [Featherston 2005], and [Clifton *et al*. 2006]; [Cowart 1997] is a user-friendly handbook.

Unfortunately, full-fledged experimental syntax is complex, forcing the researcher to spend considerable time on work that is not theoretically very interesting. Fortunately, the complexity of an experiment need only be proportional to the subtlety of the effect it is trying to detect. Most judgments are very clear (perhaps because a grammar must be shared by a speech community, and hence must be "obvious" enough to learn), and so are reliably detected even with traditional "trivially simple" methods. Very subtle or variable judgments, or hypotheses involving gradient degrees of acceptability or interactions between grammar and processing, may require full-fledged experimental methods. However, in the large area in between, a compromise seems appropriate, where methods are powerful enough to yield statistically valid results, yet are simple enough to apply quickly. This is where MiniJudge comes in.

MiniJudge [Myers 2007a] is a family of software tools designed to help theoretical syntacticians design, run, and analyze linguistic judgment experiments quickly and painlessly. Though MiniJudge experiments are small-scale experiments, testing the minimum number of speakers and sentences in the shortest amount of time, they use statistical techniques designed to maximize interpretive power from small data sets. In this paper, I first define more precisely what makes a MiniJudge experiment small-scale. Then, I walk through a sample MiniJudge experiment on Chinese. Finally, I reveal MiniJudge's inner workings, which involve some underused or novel statistical techniques. The most updated implementation of MiniJudge is MiniJudgeJS, which is written in JavaScript, HTML, and the statistical language R [R Development Core Team 2007]. It has been tested most extensively in Firefox for Windows XP, but also seems to work properly in Internet Explorer and Opera in Windows, Firefox for Linux (though line breaks are not handled properly in R for Linux), and Firefox, Opera, and Safari for Macintosh. There is also a Java implementation called MiniJudgeJava [Chen *et al.* 2007] with somewhat different internal algorithms and interface, but which otherwise works the same as the JavaScript version described in this paper.

2. Small-Scale Experimental Syntax

Experimental syntax (at least the type carried out in laboratories) generally adheres rather closely to conventions developed in psycholinguistics: multiple stimuli and subjects (naive ones rather than the bias-prone experimenters themselves), factorial designs (where materials represent all possible combinations of the experimental factors, to avoid confounds and make it possible to study interactions between factors), filler items (to prevent subjects from guessing which materials are the theoretically crucial ones), counterbalancing (so no subject is presented with "minimal pairs" differing only in theoretically relevant factors), continuous response measures (*e.g.*, open-ended judgment scales, to permit the use of standard statistical techniques like the analysis of variance, or ANOVA), and statistical analysis (to determine how unlikely the obtained results were to have occurred by chance alone). Together, these conventions can make the designing, running, and analysis of syntax experiments quite time-consuming and intimidating to the novice, especially if the experiment ends up merely reconfirming results already suspected from informally collected judgments.

In a small-scale judgment experiment, however, only the most essential of these conventions are maintained, as summarized in Table 1.

Table 1. Key characteristics of small-scale experimental syntax

Very few sentence sets (about 10)	No fillers
Very few (naive) speakers (about 10-20)	No counterbalancing of sentence lists
Maximum of two binary factors	Random sentence order
Binary *yes*/*no* judgments	Order treated as a factor in the statistics

The very small number of sentence sets and speakers (in comparison with the typical psycholinguistics experiment) means that experiments can be designed and conducted quite quickly. Statistical power need not be sacrificed, since, as explained below, the statistical analysis uses all of the raw data; hence an experiment with ten speakers judging ten sentence pairs yields 200 distinct observations. Restricting to two binary factors also speeds up experimental design, and reflects quite well the sorts of designs implicit in most actual syntactic research; an example demonstrating this is given below. Binary *yes/no* judgments are inherently less information-rich than judgments on a continuous scale, but they are generally easier for naive subjects to provide (see, *e.g.*, [Snyder 2000]); unclear cases can simply be responded to with an arbitrary guess (which may feel random, but rarely is). Though binary judgments are the default when judgments are collected informally, they are often avoided when experimenters intend to analyze their results statistically, one reason being that the most familiar statistical techniques (like ANOVA) are designed for continuous data. Rather than adjusting the judgment conventions to suit the statistics, MiniJudge adjusts the statistics to suit the judgment conventions of actual practicing syntacticians, adopting a recently developed

method designed specifically for binary response measures collected across both subjects and materials (see 4.2.1).

The lack of fillers and counterbalancing means that subjects have more opportunities to guess the purpose of the experiment than is typically tolerated in psycholinguistics, but the effect of any biases that may result is limited due to the treatment sentence order. First, as is standard in psycholinguistic experiments, materials are presented in random order, since by far the most powerful (hence, annoying) bias in linguistic responses is memory of recently processed forms. Second, going beyond standard practice, MiniJudge is capable of ignoring in the statistics any lingering order effects (see 4.2.2). As I demonstrate below, this feature is sometimes essential to bring particularly subtle and sensitive judgment patterns up to the level of statistical significance.

For further justification of the built-in restrictions of MiniJudge, see the MiniJudge homepage [Myers 2007a].

3. Using MiniJudge

To show how MiniJudge is used, I describe a recent application of it to a morphosyntactic issue in Chinese (see [Myers 2007b] for discussion of the linguistic background). MiniJudge has also been used to run syntax experiments on English and Taiwan Sign Language, as well as to run pilots for larger studies and to help teach basic concepts in experimental design. MiniJudge can also be used for judgments experiments in pragmatics, semantics, and phonology.

3.1 Goal of the Experiment

[He 2004] presents an interesting observation about the interaction of compound-internal phrase structure and affixation of the plural marker *men* in Chinese. Part of his paradigm is shown in Table 2, where V = verb and O = object (based on his (2) & (4), pp. 2-3).

Table 2. The VOmen paradigm of He (2004)

	[+men]	[-men]
[+VO]	*zhizao yaoyan zhe men *make rumor person* PLURAL	zhizao yaoyan zhe *make rumor person*
[-VO]	yaoyan zhizao zhe men *rumor make person* PLURAL	yaoyan zhizao zhe *rumor make person*

He's analysis is not relevant here; the question is simply whether or not his observation about the judgment pattern in Table 2 is empirically correct. As a non-native speaker of Chinese, I have no intuitions myself. When I have informally asked colleagues and students to double-check the judgments, I have received a mixed response. Some looking at He's paper

seem to be influenced more by the printed star pattern than the examples themselves. Others rule out *men* or VO entirely, but this misses the point, since He's claim concerns the ungrammaticality of the VO*men* form relative to all the others. It may also be that He's generalization works for the few examples he cites, but fails in general. My goal, then, was to use MiniJudge to generate more examples to test systematically on native speakers.

3.2 The MiniJudgeJS Interface

MiniJudgeJS is simply a JavaScript-enabled HTML form. Input and output are handled by text areas; generated text includes code to run statistical analyses in R. Like the rest of the MiniJudge family, MiniJudgeJS divides the experimental process into the steps in Table 3.

Table 3. The steps used by MiniJudge

I. Design experiment	**II. Run experiment**	**III. Analyze experiment**
Choose experimental factors	Choose number of speakers	Download and install R
Choose set of prototype sentences	Write instructions for speakers	Enter raw results
Choose number of sentence sets	Print or email survey forms	Generate data file
Segment prototype set (optional)	Save schematic survey file	Save data file
Replace segments (optional)		Generate R code
Save master list of test sentences		Paste R command code into R

3.3 Designing the Experiment

A MiniJudge experiment is defined by its experimental factors. Thus, the paradigm in Table 2 is derived via two binary factors: [±VO] (VO vs. OV) and [±*men*] (with or without *men* suffixation). As noted above, He's observation doesn't relate to each factor separately, but rather to an interaction: the combination of the factor values [+VO] and [+*men*] is claimed to result in lower acceptability, relative to overall judgments for [+VO] and for [+*men*].

The next step is to enter the prototype set of sentences (a pair if one factor, a quartet if two factors). Similar to the example sets shown in syntax papers and presentations, the prototype set serves multiple purposes. Most fundamentally, it helps to make the logic of factorial experimental design intuitive for novice experimenters. Syntacticians are not always aware of the importance of contrasting sentences that differ *only* in theoretically relevant factors, or of the central role played by interactions in many syntactic claims (for further discussion of the relevance of factors and interactions in syntax experiments, see [Cowart 1997], as well as the MiniJudge main page [Myers 2007a]).

Another purpose of the prototype set is that it can be used to help generate further sentence sets that maintain the same factorial contrasts but vary in irrelevant lexical properties. In the case of the present experiment, the claim made in [He 2004] says nothing about the particular verb, object, or head that is used. Thus the judgment pattern claimed for Table 2 above should also hold for the sets shown in Table 4 below, regardless of any additional influences from pragmatics, frequency, suffixlikeness (*zhe* vs. the others), or freeness (*ren* vs. the others); the stars here represent what He should predict (lexical content for the new sets was chosen with the help of Ko Yu-guang and Zhang Ning).

Table 4. Extending the VOmen paradigm of He [2004]

	[+men]	[-men]
[+VO]	*chuanbo bingdu yuan men *spread virus person PLURAL*	chuanbo bingdu yuan *spread virus person*
[-VO]	bingdu chuanbo yuan men *virus spread person PLURAL*	bingdu chuanbo yuan *virus spread person*
[+VO]	*sheji shipin ren men *design ornaments person PLURAL*	sheji shipin ren *design ornaments person*
[-VO]	shipin sheji ren men *ornaments design person PLURAL*	shipin sheji ren *ornaments design person*

MiniJudge partly automates the process of creating new sentence sets by dividing up the prototype sentences into the largest repeating segments and replacing them with user-chosen substitutes. The prototype segments for Table 2 are shown in the first row of Table 5. The user only has to find parallel substitutes for four segments, rather than having to construct whole new sentences consistent with the factorial design (Table 5 shows the segments needed to generate the new sets in Table 4). The segmentation and set generation algorithms (see 4.1) are designed to work equally well in English-like and Chinese-like orthographies. Of course, since MiniJudge knows no human language, it sometimes makes strange errors, so users are allowed to correct its output, or even to generate new sets manually.

Table 5. Prototype segments and new segments for the VOmen experiment

Set 1 (prototype) segments:	zhizao	yaoyan	zhe	men
Set 2 segments:	chuanbo	bingdu	yuan	men
Set 3 segments:	sheji	shipin	ren	men

After the user has corrected and approved the master list of sentences, it can be saved to a file for use in reports. In the present experiment, the master list contained 48 sentences (12 sets of 4 sentences each). This is an unusually large number of sentences for a MiniJudge experiment; significant results have been found with experiments with as few as 10 sentences.

3.4 Running the Experiment

In order to run a MiniJudge experiment, the user must make three decisions. The first concerns the maximum number of speakers to test. It is possible to get significant results with as few as 7 speakers, but in the present experiment, I generated 30 surveys. As it turned out, only 18 surveys were returned.

The second decision concerns whether surveys will be distributed by printed form or by email. In MiniJudgeJS, printing surveys involves saving them from a text area and printing them with a word processor. MiniJudgeJS cannot send email automatically, so emailed surveys must be individually copied and pasted. In the present experiment, I emailed thirty students, former students, or faculty of my linguistics department who did not know the purpose of the experiment.

The final decision concerns the instructions, which the user may edit from a default. MiniJudgeJS requires that judgments be entered as 1 (*yes*) vs. 0 (*no*). Chinese instructions for the VO*men* experiment were written with the help of Ko Yu-guang.

Surveys themselves are randomized individually to prevent order confounds, as is standard in psycholinguistics. The randomization algorithm, taken from [Cowart 1997:101], results in every sentence having an equal chance to appear at any point in the experiment (by randomization of blocks), while simultaneously distributing sentence types evenly and randomly.

Each survey starts with the instructions, followed by a speaker ID number (*e.g.* "##02"), and finally the survey itself, with each sentence numbered in the order seen by the speaker. Because the speakers' surveys intentionally hide the factorial design, the experimenter must save this information separately in a schematic survey file. This file is meant to be read only by MiniJudgeJS; as an example, the first line of the schematic survey file for the present experiment is explained in Table 6.

Table 6. The structure of the schematic survey information file for the VOmen experiment

File line:	`01`	`20`	`05`	`01`	`-VO`	`-men`
Explanation:	speaker ID number	sentence ID number	set ID number	order in survey	value of first factor	value of second factor

After completed surveys have been returned, the experimenter pastes them into a text area in any order (as long as each survey still contains its ID number), and pastes the schematic survey information back into another text window. MiniJudgeJS extracts judgments from the surveys and creates a data file in which each row represents a single observation, with IDs for speakers, sentences, and sets, presentation order of sentences, factor values (1 for [+] and -1 for [-]), and judgments. As an example, the first three lines of the data file for the

VO*men* experiment are shown in Table 7.

Table 7. First three lines of data file for the VOmen experiment

Speaker	Sentence	Set	Order	VO	men	Judgment
1	20	5	1	-1	-1	1
1	45	12	2	1	1	0

3.5 Analyzing the Results

For novice experimenters, the most intimidating aspect of psycholinguistic research is statistical analysis. MiniJudge employs quite complex statistical methods that are unfamiliar even to most psycholinguists, yet hides them behind a relatively user-friendly interface. Data from a MiniJudge experiment are both categorical and repeated-measures (grouped within speakers). Currently the best available statistical model for repeated-measures categorical data is generalized linear mixed effect modeling (GLMM), which can be thought of as an extension of logistic regression (see *e.g*. [Agresti *et al*. 2000].

GLMM poses serious programming challenges, so MiniJudgeJS passes the job to R, the world's foremost free statistical package [R Development Core Team 2007]. R is an open-source near clone of the proprietary program S [Chambers and Hastie 1993], and like S, is a full-featured programming language. Its syntax is somewhere between C++ and Matlab, and, of course, it has a wide variety of built-in statistical functions, including many user-written packages. The specific R package for GLMM used by MiniJudgeJS is lme4 and its prerequisite packages [Bates and Sarkar 2007].

However, since R is a command-line program, and its outputs can be unintelligible without statistical training, MiniJudgeJS handles the interface with it. The user merely enters the name of the data file, decides whether or not to test for syntactic satiation (explained below in section 3.5.2), and pastes the code generated by MiniJudgeJS into the R window. After the last line has been processed by R, the code will either generate a warning (that the file was not found or was not formatted correctly), or, if all went well, display a simple interpretive summary report. A much more detailed technical report is also saved automatically; this report is explained, step by step for the novice user, in the MiniJudge help page (Myers 2007a).

3.5.1 A Null Result?

When the data file containing the 18 completed surveys in the VO*men* experiment was analyzed using the R code generated by MiniJudgeJS, the summary report in Figure 1 was produced along with the bar graph in Figure 2. The summary report has three parts: a table showing the number of *yes* judgments for each category (shown graphically in Figure 2), a

listing of significant patterns (if any), and a statement about whether there was any significant confound between items and factors (explained more fully in section 4.2.5).

```
Number of YES judgments for each category:

             [+V]    [-V]    Total         V = VO
      [+m]   23      74      97            m = men
      [-m]   89      163     252
      Total  112     237     349

 Significance summary (p < .05):

      The factor VO had a significant negative effect.
      The factor men had a significant negative effect.
      Order had a significant negative effect.
      There were no other significant effects.

      Items and factors were significantly confounded, so the above
      results take cross-item variability into account.
```

Figure 1. Default results summary generated by MiniJudgeJS for the VOmen experiment

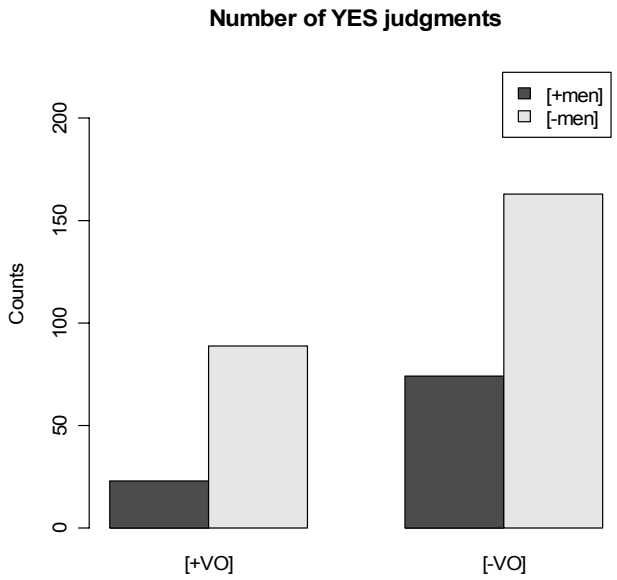


Figure 2. Default graph generated by MiniJudgeJS for the VOmen experiment

The negative effects of the [VO] and [men] factors mean that items containing VO or *men* were judged worse, on average. These patterns are also clear from the table and graph. However, as noted in 3.1, these patterns are not what the empirical claim of He [2004] is concerned with. What we expected to see was a significant interaction between [VO] and [men], but this was not found. Instead, inspection of the technical results file shows that the *p* value for the interaction was .89 for the by-speaker-only analysis and .73 for the by-speaker-and-sentence analysis, clearly non-significant ($p > .05$).

However, this is not a refutation of He's claim, but merely a null result. Indeed, the number of *yes* judgments trends in the predicted direction: for VO forms, non-*men* forms were judged better than *men* forms by a ratio of almost 4:1 (89/23 = 3.87), about twice as high as the ratio for OV forms (163/74 = 2.20). That is, it was worse to affix *men* to VO forms than to OV forms, just as He claims.

One possible cause of a null result is a confound with a nuisance variable. A clue to what this nuisance variable might be here is the significant negative effect of order, which means that judgments got worse as the experiment progressed (*i.e*. there was a rising probability of judging a form as unacceptable). This shift in judgments suggests that further analysis may be advisable, as described next.

3.5.2 Syntactic Satiation

Though MiniJudge factors out raw order effects in its default analysis, it is possible that order also *interacts* with one or more factors. Testing for interactions with continuous variables without a specific theoretical reason may make it more difficult to interpret main effects (see *e.g*. [Bernhardt and Jung 1979], but MiniJudge offers the option to test for interactions with order because it helps in the detection of syntactic satiation. Satiation is the phenomenon (known informally as "linguist's disease") in which linguistic intuitions are dulled by repeated testing, making it harder to be confident in one's judgments. MiniJudge tests for satiation by looking for interactions with order: early on, the effect of a factor is strong, but later it's weak.

[Snyder 2000] argues that satiation could provide a new window into grammar and/or processing, since different types of syntactic violations differ in whether or not they satiate. Snyder suggests two possible reasons for such differences. On the one hand, satiation may be caused by processing, not grammar, thus providing a diagnostic for performance influences on acceptability (a position taken by [Goodall 2004]). On the other hand, satiability may differ due to differences between the components of competence itself, thus permitting a new grammatical classification tool (a position taken by [Hiramatsu 2000]).

Although [He 2004] makes no predictions relating to satiation, the unexpected null result noted in section 3.5.1 suggests that it may be worthwhile trying out a more complex analysis

that includes interactions with order. Running this analysis simply involves telling MiniJudgeJS that we want to test for satiation by clicking a check box, and then pasting the newly generated code into R.

Since we chose to test satiation, MiniJudge changes the format of the graph to a line graph, as in Figure 3, which makes the overall order effect quite clear. A satiation trend is also visible in the graph, since the lines not only drop over time, but also get closer together, meaning that discrimination between sentence types weakened over the course of the experiment. Unfortunately, factoring out satiation doesn't result in any change in the main report, which comes out the same as the earlier one shown in Figure 1: no significant interaction between [VO] and [men].

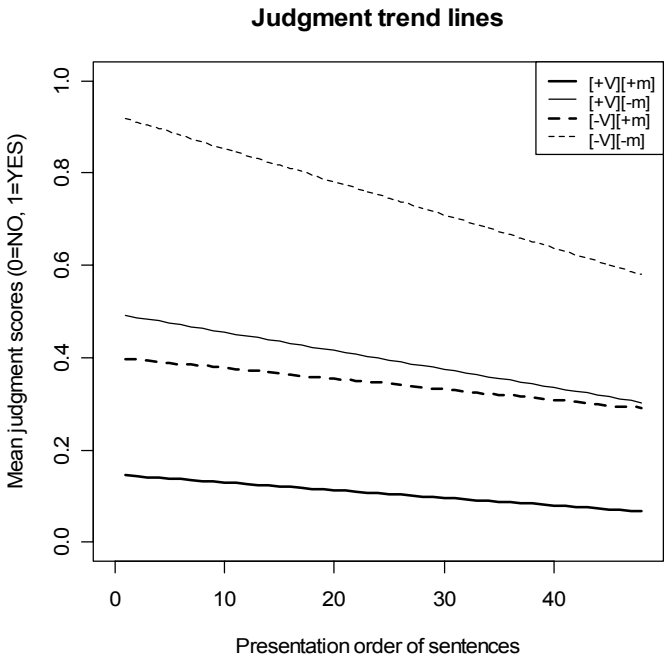


Figure 3. Graph generated by MiniJudgeJS when testing for satiation

The technical report for this analysis, automatically saved under a different name from the earlier one, clarifies what happened. The summary report gives the analysis that takes both cross-speaker and cross-sentence variability into account, since this model had statistically better coverage of the data, as indicated by the statement in the summary report that "items and factors were significantly confounded." This analysis only shows marginally significant satiation of the VO*men* effect (p = .08), and the VO*men* effect itself shows p = 0.17. However, in the less stringent, but still meaningful, by-speakers-only analysis, factoring out satiation did make the interaction between the factors [VO] and [men] significant (p = .023), and this

analysis also shows a three-way interaction between [VO], [men] and order (p = .016), that is, satiation of the VO*men* effect.

This experiment, thus, not only provided reliable evidence in favor of the empirical claim made by [He 2004], though only in the less stringent by-speakers analysis. It also revealed three additional patterns not reported by He: overall lower acceptability for VO forms relative to OV forms, overall lower acceptability of *men* forms, and the satiability of the VO*men* effect. Detecting satiation, and the VO*men* effect it obscured, depended crucially on the use of careful experimental design and statistical analysis, and would have been impossible to confirm using traditional informal methods. Despite this power, the MiniJudge experiment was designed, run, and analyzed within a matter of days (with most of the delay due to tardy subject replies), rather than the weeks required for full-fledged experimental syntax.

4. The Inner Workings

MiniJudgeJS, as with MiniJudgeJava and all future versions in the MiniJudge family, is free and open source. The JavaScript and R code can be modified freely, and both are heavily commented on to make them easier to follow. In this section I give overviews of the programming relating to material generation and statistical analysis.

4.1 Material Generation

As described in section 3.3, MiniJudgeJS can assist with the generation of additional sentence sets. This involves two major phases: segmenting the prototype sentences into the largest repeated substrings, and substituting new segments for old segments in the new sentence sets.

The first step is to determine whether the prototype sentences contain any spaces. If they do, words are treated as basic units, and capitalization is removed from the initial word and any sentence-final punctuation mark is also set aside (for adding again later). If there are no spaces (as in Chinese, or in a phonology or morphology experiment involving single words), characters are treated as basic units and there is no capitalization adjustment. Next, the boundaries between prototype sentences are demarcated to indicate that cross-sentence strings can never be segments. The algorithm for determining other segment boundaries requires the creation of a lexicon containing all unique words (or characters) in the prototype corpus. If the algorithm detects that items from the corpus and from the lexicon match only if one of the items is lowercase, this item is recapitalized. Versions of the prototype sentences with "word-based" capitalization are later used when old segments are replaced by new ones.

The most crucial step in the segmentation algorithm is to check each word (or character) in the lexicon to determine whether or not it has at least two neighbors on the same side in the corpus. For example, suppose the prototype set consists of the sentences "A dog loves the cat.

The cat loves a dog." The lexical item "loves" has two neighbors on the left: "dog" and "cat". Thus a segment boundary should be inserted to the left of "loves" in the corpus. Similarly, the right neighbor of "loves" is sometimes "the" and sometimes "a"; hence "loves" will be treated as a whole segment. By contrast, the lexical item "cat" always has the same item to its left (once sentence-initial capitalization is removed): "the". Similarly, the right neighbor of "the" is always "cat". Thus "the cat" will be treated as a segment, and the same logic applies to "a dog". The prototype segments are thus "a dog", "loves", "the cat".

The final phase involves substituting the user-chosen new segments for the prototype segments using JavaScript's built-in regular expression functions.

4.2 Statistical Analysis

The statistical analyses conducted by MiniJudgeJS involve several innovations: the use of GLMM, the inclusion of order and interactions with order as factors, the use of JavaScript to communicate with R, the use of R code to extract key values from R's technical output so that a simple report can be generated, and the use of R code to compare by-subject and by-subject-and-item analyses to decide whether the latter is really necessary. In this section I describe each of these innovations in turn.

4.2.1 GLMM

As explained in section 3.5, generalized linear mixed effect modeling (GLMM) is conceptually akin to logistic regression, which is at the core of the sociolinguistic variable-rule analyzing program VARBRUL and its descendants [Mendoza-Denton *et al*. 2003], but unlike logistic regression, GLMM regression equations also include random variables (*e.g*., the speakers); see [Agresti *et al*. 2000]. One major advantage of a regression-based approach is that no data is thrown away as it is when by-subject and by-item averages are analyzed in separate ANOVAs, as is standard in psycholinguistics (see 4.2.5). Moreover, since each observation is treated as a separate data point, GLMM is usually not affected much by missing data as long as it is missing non-systematically (this is why participants in MiniJudge experiments are requested to judge *all* sentences, guessing if they're not sure).

Though GLMM is the best statistical model currently available for repeated-measures categorical data, it does have some limitations. First, R's implementation of GLMM tests significance using *z* scores, which are most reliable if the number of observations is greater than 100 or so, but in actual practice, 100 judgments are trivial to collect (*e.g*. 5 speakers judging 10 sentence pairs). Second, like regression in general, GLMM assumes that the correlation between the dependent and independent variables is not perfect, so it is paradoxically unable to confirm the significance of perfect correlations. Third, like logistic

regression (but unlike ANOVA or ordinary regression), it is impossible to calculate GLMM coefficients and *p* values exactly; they can only be estimated. Unfortunately, the best way to estimate GLMM values is extremely complicated and slow, so R uses "simpler" yet less accurate estimation methods. Currently, R provides two options for estimating GLMM coefficients: the faster but less accurate penalized quasi-likelihood approximation, and the slower but more accurate Laplacian approximation. MiniJudgeJS uses the latter.

The function in the `lme4/Matrix` packages used for GLMM is `lmer`, which can also handle linear mixed-effect modeling (*i.e*. repeated-measures linear regression). The syntax is illustrated in Figure 3, which shows the commands used to run the final analyses described above in section 3.5.2. "Factor1" and "Factor2" are variables whose values are set in the R code to represent the actual factors. The use of categorical data is signaled by setting the distribution family to "binomial". The name of the loaded data file is arbitrarily called "minexp" (for MiniJudge experiment). The first function treats only subjects as random, while the second function treats both subjects and items as random. The choice to test for satiation or not is determined by the user; based on this choice, JavaScript generates different versions of the R code. The choice to run one-factor or two-factor analyses is determined by the R code itself by counting the number of factors in the data file. Both analyses in Figure 4 are always run, then compared with another R function described in 4.2.5.

```
glmm1 = lmer(Judgment ~ Factor1 * Factor2 * Order + (1|Speaker),
  data = minexp, family = "binomial", method = "Laplace")
glmm2 = lmer(Judgment ~ Factor1 * Factor2 * Order + (1|Speaker)
  + (1|Sentence), data = minexp, family = "binomial", method =
  "Laplace")
```

Figure 4. R commands for GLMM when testing satiation in a two-factor experiment

4.2.2 Order as a Factor

MiniJudgeJS includes order as a factor whether or not the user tests for satiation to compensate for the fact that MiniJudge experiments use no counterbalanced lists of sentences across subgroups of speakers. List counterbalancing is used in full-fledged experimental syntax so that speakers don't use an explicit comparison strategy when judging sentences from the same set (a comparison strategy may create an illusory contrast or have other undesirable consequences). However, comparison can only occur when the second sentence of a matched pair is encountered. If roughly half of the speakers get sentence type [+F] first and half get [-F] first, then on average, judgments for [+F] vs. [-F] are only partially influenced by a comparison strategy. The comparison strategy (if any) will be realized as an order effect: early

judgments (when comparison is impossible) will be different from later judgments. Thus, factoring out order effects in the statistics serves roughly the same purpose as counterbalanced lists.

4.2.3 JavaScript as an R interface

JavaScript is much more powerful than many programmers realize. In fact, a key inspiration for MiniJudgeJS was the Logistic Regression Calculating Page [Pezzullo 2005], a JavaScript-enabled HTML file written by John C. Pezzullo. Using only basic platform-universal JavaScript, the page collects data, reformats it, estimates logistic regression coefficients via a highly efficient maximum likelihood estimation algorithm, and generates chi-square values and *p* values. Thus, a JavaScript-only version of MiniJudgeJS is conceivable, without any need to pass work over to R.

Currently, however, in MiniJudgeJS, the role of JavaScript in the statistical analysis is mainly as a user-friendly GUI. Since the statistics needed for a MiniJudge experiment are highly standardized, very little input is needed from the user, but the potential to use JavaScript to interface with R in more flexible ways is there. This would help fix a major limitation with R, which has a command-line interface that is quite intimidating for novice users along with online help that leaves a lot to be desired (cf. [Fox 2005]).

Of course, JavaScript has its own limitations, the most notable of which are the built-in security constraints that prevent JavaScript from being able to read or write to files, or to communicate directly with other programs. For example, it's impossible to have JavaScript run R in the background, to save users the bother of copying and pasting in R code. This is why we developed MiniJudgeJava as well, though, in its current version, it still requires the user to interface with R by pasting in code.

4.2.4 R Code to Simplify Output

GLMM is a high-powered statistical tool, unlikely to be used by people who don't already have a strong background in statistics; therefore, the outputs generated by R are not understandable without such a background. Since MiniJudge is intended for statistical novices, extra programming is needed to translate R output into plain language. For MiniJudgeJS, the most crucial portion of R's output for GLMM is the matrix containing the regression coefficient estimates and *p* values, like that shown in Figure 5 (from the VO*men* experiment, without testing for satiation). The trick is to extract the estimates (the signs of which provide information about the nature of the pattern) and the *p* values (which indicate significance) in order to generate a simple summary containing no numbers at all.

```
                  Estimate    Std. Error  z value    Pr(>|z|)
(Intercept)       -0.198279   0.392239    -0.506     0.6132
Factor1           -1.125097   0.252556    -4.455     8.40e-06
Factor2           -1.420005   0.253253    -5.607     2.06e-08
Order             -0.019289   0.007518    -2.566     0.0103
Factor1:Factor2   0.087524    0.251891    0.347      0.7282
```

Figure 5. Output generated by `lmer` for the VOmen experiment without testing for satiation

The current version of MiniJudgeJS extracts values from the `lmer` output by "sinking" `lmer`'s displayed output to an offline file, and then reading this file back in as a string (the offline file becoming the permanent record of the detailed analysis). The string is then searched for the string "(Intercept)" which always appears at the upper left of the value matrix. The coefficient is the first value to the right of the left-most column, and the *p* value is the fourth value (skipping "<", if any).

If the *p* value associated with a factor or interaction is less than 0.05, a summary line is generated that gives the actual factor name and the sign of the estimate, as in Figure 1 above. The R code generates the summary table and bar graph counting the number of *yes* judgments for each category (see Figures 1 and 2) directly from the data file itself. When satiation is tested, the line graph (as in Figure 3) is created by computing the mean judgment values (*i.e.* proportion of 1 judgments) across speakers with each order value (*i.e.* 1, 2, ...), separately for each item type (as defined by the experimental factors), and then plotting linear regression lines for each item type.

4.2.5 By-Subject and By-Item Analyses

MiniJudgeJS runs both by-subject and by-subject-and-item analyses, but it reports only the first in the main summary unless it finds that the more complex analysis is really necessary. This approach differs from standard psycholinguistic practice, where both by-subject and by-item analyses are always run. A commonly cited reason for always running a by-item analysis is that it is required to test for generality across items, just as a by-subject analysis tests for generality across subjects. However, this logic is based on a misinterpretation of [Clark 1973], which is the paper usually cited as justification.

First, it is wrong to think that by-item analyses check to see if any item behaves atypically (*i.e.* is an outlier). For parametric models like ANOVA, it is quite possible for a single outlier to cause an illusory significant result, even in a by-item analysis (categorical data analyses like GLMM don't have this weakness). To test for outliers, there's no substitute for checking the individual by-item results manually. MiniJudge helps with this by reporting

the by-sentence rates of *yes* judgments in a table saved as part of the offline analysis file; items with unusually low or high acceptability relative to others of their type stand out clearly. In the case of the VO*men* experiment, this table did not seem to show any outliers.

The second problem with the standard justification for performing obligatory by-item analyses, as [Raaijmakers *et al*. 1999] emphasize, is that the advice given in [Clark 1973] actually applies only to experiments without matched items, such as an experiment comparing a random set of sentences with transitive verbs ("eat", etc.) with a random set of sentences with unrelated intransitive verbs ("sleep", etc.). Such sentences will differ in more than just the crucial factor (transitive vs. intransitive), so, even if a difference in judgments is found, it may actually relate to uninteresting confounded properties (*e.g*. the lexical frequency of the verbs). However, if lexically matched items are used, as in the VO*men* experiment, there is no such confound, since items within each set differ only in terms of the experimental factor(s). If items are sufficiently well matched, taking cross-item variation into account won't make any difference in the analysis (except to make it much more complicated), but if they are not well matched, ignoring the cross-item variation will result in misleadingly low *p* values.

Nevertheless, if we only computed models that take cross-item variation into account, we might lose useful information. After all, a high *p* value does not necessarily mean that there is no pattern at all, just that we have failed to detect the pattern. Thus, it may be useful to know if a by-speaker analysis is significant even if the by-speaker-and-sentence analysis is not. Such an outcome could mean that the significant by-speaker result is an illusion due to an uninteresting lexical confound, but it could instead mean that if we do a better job matching the items in our next experiment, we will be able to demonstrate the validity of our theoretically interesting factor. Moreover, it is quite difficult to compute GLMM models with two random variables, making such models somewhat less reliable than those with only one random variable. Just in the last year, the `lme4` package in R has been upgraded, so that the `lmer` function now gives different results for by-subjects-and-items analyses than it did when MiniJudge was first developed. Due to concerns like these, MiniJudge runs both types of analyses and only chooses the by-subjects-and-items analysis for the main report if a statistically significant confound between factors and items is detected. The full results of both analyses are saved in an off-line file, along with the results of the statistical comparison of them.

The R language makes it quite easy to perform this comparison, since the model in which only speakers are treated as random is a special case of the model in which both speakers and sentences are treated as random. This means the two GLMM models can be compared by a likelihood ratio test using ANOVA [Pinheiro and Bates 2000]. As with the output of the `lmer` function, the output of the `lme4` package's `anova` function makes it difficult to extract *p* values, so again the output is "sunk" to the offline analysis file to be read back in as a string.

Only if the *p* value is below 0.05 is the more complex model taken as significantly better. If the *p* value is above 0.2, MiniJudgeJS assumes that items and factors are not confounded and reports only the by-subjects-only analysis in the main summary. Nevertheless, MiniJudgeJS, erring on the side of caution, gives a warning if $0.2 > p > 0.05$. In any case, both GLMM analyses are available for inspection in the offline analysis file. Each analysis also includes additional information, generated by `lmer`, that may help determine which analysis is really more reliable, including variance of the random variables and the estimated scale (compared with 1); these details are explained in the MiniJudge help page.

In the case of the VO*men* experiment, the comparison of the two models showed that the by-subjects-only model was sufficient, unsurprisingly, given that the materials were almost perfectly matched, and that the items table showed no outliers among the sentence judgments.

The final problem with the standard justification for automatic by-item analyses is one that even [Raaijmakers *et al*. 1999] fail to point out. Namely, since repeated-measures regression models make it possible to take cross-speaker and cross-sentence variation into account at the same time, without throwing away any data, they are superior to standard models like ANOVA. To learn more about how advances in statistics have made some psycholinguistic traditions obsolete, see [Baayen 2004].

5. Conclusions

MiniJudge, currently implemented as MiniJudgeJS and MiniJudgeJava, is software for theoretical syntacticians who want a reliable and easy way to collect and interpret judgments consistent with the key methodological principles of experimental cognitive science. MiniJudge is limited in some ways, in particular in how it interfaces with R, though, in ongoing work, we are developing efficient code to compute GLMM within JavaScript or Java itself. Nevertheless, even in its current version, MiniJudge is quite easy to use, as testing by my students has demonstrated, and powerful enough to detect theoretically interesting patterns with very little data. Behind this power is original programming and statistical techniques. Finally, MiniJudge is an entirely free, open-source program (as will be all future versions). Anyone interested is invited to try it out and contribute to its further development.

Acknowledgements

This research was supported by National Science Council (Taiwan) grant NSC 94-2411-H-194-018. MiniJudgeJS is co-copyrighted by National Chung Cheng University. Experimental or programming help came from my research assistants Ko Yu-guang, Chen Tsung-yin, and Yang Chen-Tsung. The students in my spring 2006 class *Competence & Performance* helped test MiniJudgeJS and made useful suggestions. John C. Pezzullo and Harald Baayen also provided helpful information on programming and statistical matters. An

earlier version was presented at ROCLING 18 (Hsinchu, Taiwan, September, 2006), and I thank conference reviewers and audience members for their comments. Of course I am solely responsible for any mistakes.

References

Agresti, A., J. G. Booth, J. P. Hobert, and B. Caffo, "Random-Effects Modeling of Categorical Response Data," *Sociological Methodology*, 30, 2000, pp. 27-80.

Baayen, R. H., "Statistics in Psycholinguistics: A Critique of Some Current Gold Standards," *Mental Lexicon Working Papers*, vol. 1, ed. by G. Libben and K. Nault, University of Alberta, Canada, 2004, pp. 1-45.

Bates, D., and D. Sarkar, lme4: Linear Mixed-Effects Models Using S4 Classes, version 0.99875-7, http://cran.r-project.org/src/contrib/Descriptions/lme4.html, 2007.

Bernhardt, I., and B. S. Jung, "The Interpretation of Least Squares Regression with Interaction or Polynomial Terms," *The Review of Economics and Statistics*, 61(3), 1979, pp. 481-483.

Chambers, J. M., and T. J. Hastie, *Statistical Models in S*, Chapman & Hall, New York, 1993.

Chen, T.-Y., C.-T. Yang, and J. Myers, MiniJudgeJava, version 0.9.9, http://www.ccunix.ccu.edu.tw/~lngproc/MiniJudge.htm, 2007.

Chomsky, N., *Aspects of the Theory of Syntax*, The MIT Press, Cambridge, MA, 1965.

Clark, H., "The Language-As-Fixed-Effect Fallacy: A Critique of Language Statistics in Psychological Research," *Journal of Verbal Learning and Verbal Behavior*, 12, 1973, pp. 335-359.

Clifton, Jr., C., G. Fanselow, and L. Frazier, "Amnestying Superiority Violations: Processing Multiple Questions," *Linguistic Inquiry*, 37(1), pp. 51–68.

Cowart, W., *Experimental Syntax: Applying Objective Methods to Sentence Judgments*, Sage Publications, London, 1997.

Featherston, S., "*That*-Trace in German," *Lingua*, 115(9), 2005, pp. 1277-1302.

Fox, J., "The R Commander: A Basic-Statistics Graphical User Interface to R," *Journal of Statistical Software*, 14(9), 2005, pp. 1-42.

Goodall, G., "On the Syntax and Processing of Wh-Questions in Spanish," *Proceedings of the 23rd West Coast Conference on Formal Linguistics*, 2004, Davis, California, pp. 101-114.

He, Y., "The Words-and-Rules Theory: Evidence from Chinese Morphology," *Taiwan Journal of Linguistics*, 2(2), pp. 1-26.

Hiramatsu, K., Accessing Linguistic Competence: Evidence from Children's and Adults' Acceptability Judgements, PhD thesis, University of Connecticut, Storrs, 2000.

Mendoza-Denton, N., J. Hay, and S. Jannedy, "Probabilistic Sociolinguistics: Beyond Variable Rules," *Probabilistic Linguistics*, ed. by R. Bod, J. Hay and S. Jannedy, The MIT Press, Cambridge, MA, 2003, pp. 97-138.

Myers, J., MiniJudgeJS, Version 1.0, www.ccunix.ccu.edu.tw/~lngproc/MiniJudgeJS.htm, 2007a.

Myers, J., "Generative Morphology as Psycholinguistics," *The Mental Lexicon: Core Perspectives*, ed. by G. Jarema and G. Libben, Elsevier, Amsterdam, pp. 105-128.

Penke, M., and A. Rosenbach, "What Counts as Evidence in Linguistics? An Introduction," *Studies in Language*, 28(3), 2004, pp. 480-526.

Pezzullo, J. C., Logistic Regression Calculating Page, version 05.07.20, http://statpages.org/logistic.html, 2005.

Pinheiro, J. C., and D. M. Bates, *Mixed-Effects Models in S and S-Plus*, Springer, Berlin, 2000.

R Development Core Team, R: A Language and Environment for Statistical Computing, http://www.R-project.org.

Raaijmakers, J. G. W., J. M. C. Schrijnemakers, and F. Gremmen, "How to Deal with 'the Language-As-Fixed-Effect Fallacy': Common Misconceptions and Alternative Solutions," *Journal of Memory and Language*, 41, 1999, pp. 416-426.

Schütze, C. T., *The Empirical Base of Linguistics: Grammaticality Judgments and Linguistic Methodology*, University of Chicago Press, Chicago, 1996.

Snyder, W., "An Experimental Investigation of Syntactic Satiation Effects," *Linguistic Inquiry*, 31, 2000, pp. 575-582.

Sorace, A., and F. Keller, "Gradience in Linguistic Data," *Lingua*, 115, 2005, pp. 1497-1524.

Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 195-216

Improve Parsing Performance by Self-Learning

Yu-Ming Hsieh*, Duen-Chi Yang*, and Keh-Jiann Chen*

Abstract

There are many methods to improve performance of statistical parsers. Resolving structural ambiguities is a major task of these methods. In the proposed approach, the parser produces a set of *n*-best trees based on a feature-extended PCFG grammar and then selects the best tree structure based on association strengths of dependency word-pairs. However, there is no sufficiently large Treebank producing reliable statistical distributions of all word-pairs. This paper aims to provide a self-learning method to resolve the problems. The word association strengths were automatically extracted and learned by parsing a giga-word corpus. Although the automatically learned word associations were not perfect, the constructed structure evaluation model improved the bracketed *f*-score from 83.09% to 86.59%. We believe that the above iterative learning processes can improve parsing performances automatically by learning word-dependence information continuously from web.

Keywords: Parsing, Word association, Knowledge Extraction, PCFG, PoS Tagging, Semantic.

1. Introduction

How to solve structural ambiguity is an important task in building a high-performance statistical parser, particularly for Chinese. Since Chinese is an analytic language, words can play different grammatical functions without inflection. A great deal of ambiguous structures would be produced by parsers if no structure evaluation were applied. There are three main steps in our approach that aim to disambiguate the structures. The first step is to have the parser produce *n*-best structures. Second, we extract word-to-word associations from large corpora and build semantic information. The last step is to build a structural evaluator to find the best tree structure from the *n*-best candidates.

There have been some approaches proposed in the past to resolve structure ambiguities. For instance:

* Institute of Information science, Academia Sinica, Taipei, Taiwan
E-mail: {morris, ydc, kchen}@iis.sinica.edu.tw

[Received February 7, 2007; Revised July 2, 2007; Accepted July 9, 2007]

Adding on lexical dependencies. Collins [1999] solves structural ambiguity by extracting lexical dependencies from Penn WSJ Treebank and applying dependencies to the statistic model. Lexical dependency (or Word-to-word association, WA) is one type of semantic information. It is a current trend to add on semantic related information in traditional parsers. Some incorporate word-to-word association in their parsing models, such as the Dependency Parsing in Chen *et al*. [2004]. They take advantage of statistical information of word dependency in the parsing process to produce dependency structures. However, word association methods suffer low coverage when lacking very large tree-annotated training corpora while checking dependency relationships between word pairs.

Adding on word semantic knowledge where CiLin and HowNet information are used in the statistic model in the experiment [Xiong *et al*. 2005]. Their results work to solve common parsing mistakes efficiently.

Using a re-annotation method in grammar rules. Johnson [1998] thinks that re-annotating each node with the category of its parent category in Treebank is able to improve parsing performance. Klein *et al*. [2003] proposes internal, external, and tag-splitting annotation strategies to obtain better results.

Building an evaluator. Some people re-rank the structure values and find the best parse [Collins 2000; Charniak *et al*. 2005]. At first, the parser produces a set of candidate parses for each sentence. Later, the re-ranker finds the best tree through relevance features. The performance is better than without the re-ranker.

This paper is going to show a self-learning method to produce imperfect (due to errors produced by automatic parsing) but unlimited amount of word association data to evaluate the *n*-best trees produced by a feature-extended PCFG grammar. The parser with this WA evaluation is considerably superior to those without the evaluation.

The organization of the paper is as follows: Section 2 describes how to generate *n*-best trees in a simple way. In Section 3, we account for building word-to-word association and a primitive semantic class as well. As to the design of the evaluating model, our probability model, coordination of rule probability, and word association probability are presented in Section 4. In Section 5, we discuss and explain the experimental data and results. Ambiguities of PoS are to be considered in a practical system. Section 6 deals with further experiments on

automatic tagging with PoS. Finally, we offer concluding remarks in Section 7.

2. Feature Extension of PCFG Grammars for Producing the *N*-best Trees

It is clear that Treebanks [Chen *et al*. 2003] provide not only instances of phrasal structures and word dependencies but also their statistical distributions. Recently, probabilistic preferences for grammar rules and feature dependencies were incorporated to resolve structure-ambiguities and had great improvement on parsing performance. However, the automatically extracted grammars and feature-dependence pairs suffer the problem of low coverage. We proposed different approaches to solve these two different types of low coverage problems. For the low coverage of extracted grammar, a linguistically-motivated grammar generalization method is proposed [Hsieh *et al*. 2005]. The low coverage of word association pairs is resolved by a self-learning method of automatic parsing and extracting word dependency pairs from very large corpora.

The linguistically-motivated generalized grammars are derived from probabilistic context-free grammars (PCFG) by right-association binarization and feature embedding. The binarized grammars have better coverage than the original grammars directly extracted from Treebank. Features are embedded into the lexical and phrasal categories to improve the precision of generalized grammar. The important features adopted in our grammar are described in the following:

Head (Head feature):	The PoS of phrasal head will propagate all intermediate nodes within the constituent.
Example:	S(NP(Head:Nh:他)|S'$_{\text{-Head:VF}}$(Head:VF:叫|S'$_{\text{-Head:VF}}$(NP(Head:Nb:李四)| VP(Head:VC:撿| NP(Head:Na:皮球)))))
Linguistic motivations:	To constrain the sub-categorization frame.
Left (Leftmost feature):	The PoS of the leftmost constitute will propagate one-level to its intermediate mother-node only.
Example:	S(NP(Head:Nh: 他)|S'$_{\text{-Head:VF}}$(Head:VF: 叫 |S'$_{\text{-NP}}$(NP(Head:Nb: 李四)| VP(Head:VC:撿| NP(Head:Na:皮球)))))
Linguistic motivation:	To constrain linear order of constituents.
Head 0/1 (Existence of phrasal head):	If phrasal head exists in intermediate node, the nodes will be marked with feature 1; otherwise 0.
Example:	S(NP(Head:Nh: 他)|S'$_{\text{-1}}$(Head:VF: 叫 |S'$_{\text{-0}}$(NP(Head:Nb: 李四)|VP(Head:VC:撿| NP(Head:Na:皮球)))))
Linguistic motivation:	To enforce unique phrasal head in each phrase.

There are two functions of applying the embedded features: one is to increase the precision of the grammar and the other is to produce more candidate parse structures. With features embedded in phrasal categories, PCFG parsers are forced to produce varieties of different possible structures[1]. In order to achieve a better *n*-best oracle performance (*i.e*. the ceiling performance achieved by picking the best structure from *n* bests), we designed some different feature-embedded grammars and try to find a grammar with the better *n*-best oracle performance. For instance, "S(NP(Head:Nh: 他)|Head:VF: 叫 | NP(Head:Nb: 李 四)| VP(Head:VC:撿| NP(Head:Na:皮球)))". The explanations of feature sets are as follow.

Rule type-1:

Intermediate node: add on "Left and Head 1/0" features.

Non-intermediate node: if there is only one member in the NP, add on "Head" feature.

Example: S(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S'$_{\text{-Head:VF-1}}$(Head:VF:叫|S'$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:皮球)))))

Rule type-2:

Intermediate node: add on "Left and Head 1/0" features.

Non-intermediate node: add on "Head and Left" features, if there is only one member in the NP, add on "Head" feature.

Example: S$_{\text{-NP-Head:VF}}$(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S'$_{\text{-Head:VF-1}}$(Head:VF:叫 |S'$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP$_{\text{-Head:VC}}$(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:皮球)))))

Rule type-3:

Intermediate node: add on "Left, and Head 1/0" features.

Top-Level node: add on "Head and Left" features. (see example of S$_{\text{-NP-Head:VF}}$)

Non-intermediate node: if there is only one member in the NP, add on "Head" feature.

Example: S$_{\text{-NP-Head:VF}}$(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S'$_{\text{-Head:VF-1}}$(Head:VF:叫 |S'$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:皮球)))))

[1] The parser adopts an Earley's Algorithm. It is a top-down left-to-right algorithm. So, in parts that have the same non-terminals, we keep only the best structure after pruning, to reduce the load of calculation and thus fasten the parsing speed. Therefore, if we add different features in the Top-Level rules, we'll get more results.

Rules and their statistical probabilities are extracted from the transformed structures. The grammars are derived and trained from Sinica Treebank[2]. Sinica Treebank contains 38,944 tree-structures and 230,979 words. Table 1 shows the number of rule types in each grammar and Table 2 shows their 50-best oracle bracketed *f*-scores on three sets of testing data. The three sets of testing data used in our experiments represent "moderate", "difficult", and "easy" scale of Chinese language respectively. Black [1991] proposed two structural evaluating systems in 1991; the more strictly based is named PARSEVAL, and the less strictly based is crossing. We adopt PARSEVAL measures to evaluate the bracketed *f*-score. The formula represents as follows:

$$\text{bracketed precision (BP)} = \frac{\#\text{ brack correct constituents in parser's parse of testing data}}{\#\text{ bracket constituents in parser's parse of testing data}}$$

$$\text{bracketed recall (BR)} = \frac{\#\text{ brack correct constituents in parser's parse of testing data}}{\#\text{ bracket constituents in treebank's parse of testing data}}$$

$$\text{bracketed } f\text{-score (BF)} = \frac{BP * BR * 2}{BP + BR}$$

A bracket represents the phrasal scope. The reason we don't use a labeled *f*-score is that we aim to evaluate the phrasal scope, rather than the effect brought by the phrasal category. For example, the dependency information is much more related to the structure.

Table 1. Numbers of rules for each grammar.

	Rule Type		
	Rule-1	Rule-2	Rule-3
Rule number	9,899	26,797	13,652

Table 2. The 50-best oracle performances from the different grammars.

Testing Data	Sources	Hardness	Rule type		
			Rule type-1	Rule type-2	Rule type-3
Sinica	Balanced corpus	Moderate	92.97	94.84	96.25
Sinorama	Magazine	Difficult	90.01	91.65	93.91
Textbook	Elementary school	Easy	93.65	95.64	96.81

[2] http://treebank.sinica.edu.tw/

From the above table, we can observe that the "Rule type-3" outperforms the "Rule type-1" and "Rule type-2". We adopt the approach used in Charniak *et al.* [2005] to analyze the *n*-best parse. Table 3 shows the best bracketed *f*-score values of different *n*-best parse trees. From the results, we observe that the improvement after *n*=5 is slight. Thus, the number of ambiguous candidates can be dynamically adjusted according to the complexity of input sentences. For normal sentences, we may consider to take *n*=5 in order to minimize the complexity. For long sentences or sentences with auto PoS tagging should take as large as *n*=50 to raise the ceiling of the best *f*-score.

Table 3. Oracle bracketed f-scores as a function of number n of n-best parses.

Testing Data	*n*					
	1	2	5	10	25	50
Sinica	91.88	94.39	95.91	96.17	96.25	96.25
Sinorama	86.69	90.44	92.87	93.47	93.86	93.91
Textbook	92.24	95.01	96.21	96.61	96.78	96.81

For each candidate tree, its syntactic plausibility is obtained by rule probabilities produced by PCFG parser. In addition to this, we need semantic related information to help with finding the best tree structure among candidate trees. In the next section, we will look at some methods of attaining semantic related information.

3. Auto-Extracting World Knowledge

We could extract word knowledge from Treebanks, but the availability of a very large set of trees with rich linguistic annotations has long been a problem. A cheaper way to extract word knowledge is to automatically parse a large amount of data. We believe that with good parsing performance, we could get sufficient information.

Therefore, in our experiments, we use a Gigaword Chinese corpus to extract word dependence pairs. The Gigaword corpus contains about 1.12 billion Chinese characters, including 735 million characters from Taiwan's Central News Agency (traditional characters), and 380 million characters from Xinhua News Agency (simplified characters)[3]. Word associations are extracted from the texts of Central News Agency (CNA). First we use Chinese Autotag System [Tsai *et al*. 2003], developed by Academia Sinica, to process the segmentation and PoS tagging of the texts. This system reaches a performance of 95% segmentation and 93% tagging accuracies. Then we parse each sentence[4] in the corpus and assign semantic roles to each constituent. Based on the head word information, we extract

[3] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T09

[4] An existing parser is used to produce 1-best tree of a sentence.

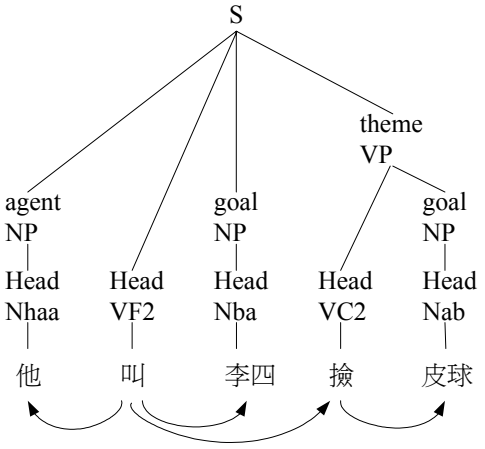
dependence word-pairs between head words and their arguments or modifiers. The following illustrates how the automatic knowledge extraction works. We input a Chinese sentence to the parser:

他 叫 李四 撿 皮球
Ta jiao Li-si jian qiu
He ask Li-si pick ball
"He asked Li-si to pick up the ball."

Here is the sentence after segmentation and PoS tagging:

他(Nh) 叫(VF) 李四(Nb) 撿(VC) 皮球(Na)

The parser analyzes the sentence structure and assigns roles to each phrase as follows. Then, word-pair knowledge of heads and their modifiers are extracted as shown in Figure 1.



Role1	PoS1	Word1	Role2	PoS2	Word2
agent[NP]	Nhaa	他	Head[S]	VF2	叫
Head[S]	VF2	叫	goal[NP]	Nba	李四
Head[S]	VF2	叫	theme[NP]	VC2	撿
Head[S]	VC2	撿	goal[NP]	Nab	皮球

Figure 1. A sample for word association extraction.

Figure 1 shows the examples of extracted word associations. “Role1/PoS1/Word1 and Role2/Role2/Word2” represent the right- and left-part of the word-pairs. “Role”, “PoS”, and “Word” here mean semantic role, part-of-speech and word respectively. To reduce the number of word association types, we transform the original word-pairs into three simplified types of the word pairs:

(a) head word on the left hand side: (H_W_C, X_W_C);

(b) head word on the right hand side: (X_W_C, H_W_C);

(c) coordinating structure: (H_W_C, H_W_C).

In the word pairs, “H” denotes Head, “W” means word, and “C” refers to the simplified PoS tag[5], “X” refers to any semantic role other than Head role. So, we get basic information of experimental data as follows:

Role1	PoS1	Word1	Role2	PoS2	Word2
X	Nh	他	H	VF	叫
H	VF	叫	X	Nb	李四
H	VF	叫	X	VC	撿
H	VC	撿	X	Na	皮球

The processes above are repeated for each new input sentence from the Gigaword corpus.

Finally, we obtain a great deal of knowledge about dependent word pairs and their association strengths. In our experiments, we have 37,489,408 sentences that are successfully parsed and contain word association information. The number of extracted word associations is 221,482,591. The extracted word to word associations that undergo structure analysis and head word assignment are not perfectly correct, but they are more informative and precise than simply taking words on the left and right hand window.

3.1 Coverage Rates of the Word Associations

Data sparseness is always a problem of statistical evaluation methods. As mentioned in the last section, we automatically segment, tag, parse and assign roles in CNA data, and then extract word associations. We test our extracted word association data in five different levels of granularities. Level-1 to Level-5 represents HWC_WC, HW_W, HC_WC, HW_C, and HC_C respectively. The 5 levels of word associations derived from Figure 1 are as follows:

[5] The simplified way please refer to CKIP 93-05 Technical Report.

Level	Type	Word Associations
Level-1	HWC_WC	(他/Nh _H/叫/VF) (H/叫/VF_李四/Nb) (H/叫/VF_撿/VC) (H/撿/VC_球/Na)
Level-2	HW_W	(他_H/叫) (H/叫_李四) (H/叫_撿) (H/撿_皮球)
Level-3	HC_WC	(他/Nh_H/VF) (H/VF_李四/Nb) (H/VF_撿/VC) (H/VC_皮球/Na)
Level-4	HW_C	(Nh_H/叫) (H/叫_Nb) (H/叫_VC) (H/撿_Nb)
Level-5	HC_C	(Nh_H/VF) (H/VF_Nb) (H/VF_VC) (H/VC_Na)

Theoretically, the precision of fine-grain level like HWC_WC is much better, but it suffers the problem of data sparseness, hence, its coverage rate is low; on the other hand, the coarse-grain level has best coverage rate but relatively low precision. This is the trade-off between precision and coverage. Therefore, we carry out a series of experiments to find a balanced measurement by linear combination of different level associations. There will be experimental results in the following sections.

Why not use HWC_W or HC_W? From our observation, we have found that these two show similar performance with HWC_WC and HC_WC respectively; therefore, we exclude them. Besides, there are some asymmetric representations, such as the use of "HW_C". They are used to raise the coverage rate in word association while not being too general.

We like to see the bi-gram coverage rates for each level of representation. After CNA producing word associations in each level, we observe the relationship between the amount of word associations and the coverage rates of the three texts: Sinica, Sinorama, and Textbook. We extracted word associations from the three data sets in each level and calculated their coverage rates.

We tested the coverage rates for 10 different size word association data, of which each was extracted from different size corpora. Figure 2 shows coverage relationships between five levels and sizes of word association data for three testing data.

Figure 2 shows that larger data increases the coverage rates, but the coverage of the fine-grained level word associations, *e.g*. Level-1 (HWC_WC), is only about 70%, which is far from saturation. Nonetheless, the coverage rate can be improved by reading more texts from the web. The coarse-grained level associations, *e.g*. Level-5 (HC_C), cover the most bi-gram categories. However, it may not be very useful, since syntactic associations which are partially embedded in the PCFG are redundant. To attain a better evaluation model, we derived new associations between semantic classes. Criteria for semantic classification are discussed in the following section.

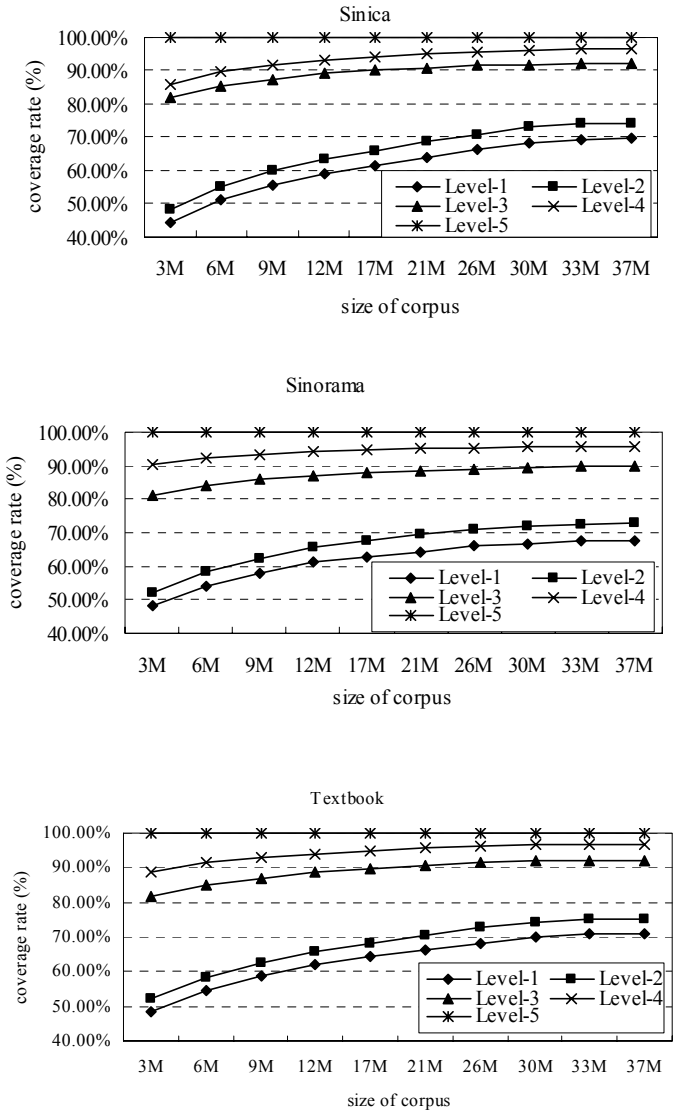


Figure 2. Coverage rates vs. size of Corpus: (a) Sinica; (b) Sinorama; (c) Textbook.

3.2 Incorporating Semantic Knowledge

For precision and coverage tradeoffs, we face a dilemma of using word or PoS category. We find that the coverage of word is low, though its precision is high; on the contrary, the coverage of PoS is too high to be discriminative. We hope to find a classification that covers enough information and is discriminative as well; that is, a classification system that falls between word and PoS category. A semantic classification is the solution.

There are many ways to classify semantic properties of words. Xiong *et al*. [2005] adopt CiLin and HowNet as their semantic classes in their experiment; however, the data sparseness is still a problem to be solved. Here, we propose a simple approach to build a semantic-class-based association strength for word pairs, which will be our Level-6 (HS_S). Semantic class information is put into Level-6 in order to get high coverage and to avoid redundant syntactic associations in other levels. Besides, it can smooth the problem of data sparseness.

The idea is to classify words into their head morpheme. It begins with the transformation of every input "WORD, POS" in the data. We adopt the affix database of high frequency verbs and nouns [Chiu *et al*. 2004] to set up noun and verb classes. There are 34,857 examples of compound words in the database. As to determinative measures (DM), we refer to the dictionary of measure words, and divide the DMs in the data into thirteen categories, according to the meanings of the measure words. The thirteen categories include general, event, length, science, approximate measures, weight, square measures, container, capacity, time, currency value, classification measures, and measures of verbs. Finally, we consult parts of speech analyses [CKIP 1993] and set up the transformation rules to transform a word-PoS pair into its semantic class. The transformation algorithm is shown at Appendix A. Take "李四, Nb" as example, its semantic class is "PersonalName(人名)" in our classification. In another instance, the semantic class of "皮球, Na" is "Na_球". The transformation rules are PoS dependent. Each PoS is referred to CKIP [1993], which explains the PoS with words and examples. We set up discriminative subcategorization on some parts-of-speech: N/P/D/A according to the distribution of PoS and word frequency. As to the verbs, we use an initial step to assign initial value. Take PoS as "A" for example, adding prefix information is more useful than using "A" alone.

Role1	PoS1	Word1	Class1	Role2	PoS2	Word2	Class2
X	Nh	他	他	H	VF	叫	叫
H	VF	叫	VF_叫	X	Nb	李四	PersonalName
H	VF	叫	VF_叫	X	VC	撿	VC_撿
H	VC	撿	VC_撿	X	Na	皮球	Na_球

The following example is the result of DM, prefix and affix, through a function in Level-6 (HS_S):

S(theme:NP(quantifier:DM:兩個|Head:Nab:人)|deontics:Dbab:能|Head:VC1:在|goal:GP(DUMMY:NP(property:Nad:人生|Head:Nad:旅途)|Head:Ng:中))

Role1	PoS1	Word1	Class1	Role2	PoS1	Word1	Class2
X	DM	兩個	general_DM	H	Na	人	Na_人
X	Na	人	Na_人	H	VCL	在	VCL_在
X	D	能	D_能	H	VCL	在	VCL_在
H	VCL	在	VCL_在	X	Na	旅途..中	Na..Ng
X	Na	人生	N_人	H	Na	旅途	Na_途
X	Na	旅途	Na_途	H	Ng	中	Location

It is necessary to discriminate syntactic head from semantic head in word association extraction of GPs and PPs. From the table above, Row 4, signified by the different color shows that "旅途" is the semantic head of the GP "旅途..中", while the word "中" is the syntactic head of the phrase.

We estimate the word association coverage rate of the Level-6 associations. From the results shown in Figure 3, the coverage rate of Level-6 is higher than Level-2, and the problem of data sparseness is indeed moderately smoothed.

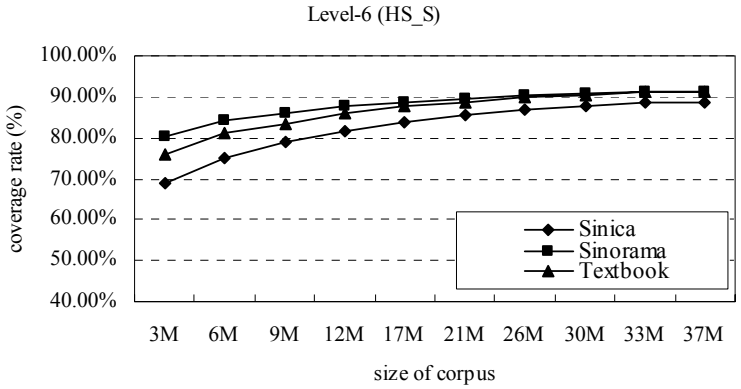


Figure 3. WA coverage rate of Level-6.

Next, we will use different levels of associations to construct an evaluation model to find the best structure among the numerous ambiguous candidates.

4. Building Evaluation Model

A sentence structure is evaluated by its syntactic and semantic plausibility. The syntactic plausibility is modeled by products of phrase rule probabilities of its syntactic tree. The semantic plausibility is modeled by the word association strengths between head words and their arguments or modifiers. For an input sentence S, the feature-embedded PCFG parser produces its n-best trees $\{y_1(s),...,y_n(s)\}$. The evaluating model finds out the best structure

according to the rule probability (syntactic) and corresponding word association probability (semantic). Rule probabilities are generated by the PCFG parser when n-best trees are produced. We will estimate word association probabilities in the following formula. In the formula, "Head" means the Head of a word association, notated as HWC, HC, or HW. "Modify" means dependent daughter, notated as WC, W, or C.

$$P(Modify \mid Head) = \frac{freq(Head, Modify)}{freq(Head)} \tag{1}$$

Data sparseness is a common problem in dealing with corpora. A minimal value σ is used to smooth data sparseness:

$$\sigma = \frac{1}{total\ number\ of\ WA\ token} = \frac{1}{221482591}$$

The evaluation value $Value(y_n(s))$ to each candidate tree *Yn(S)* is defined as:

$$\begin{aligned} &Value(y_n(s)) = \\ &\lambda * RuleValue(y_n(s)) + (1-\lambda) WAValue(y_n(s)) \end{aligned} \tag{2}$$

where $RuleValue(y_n(s))$ is the rule probability of the sentence and $WAValue(y_n(s))$ is the total word association value in different level *n*. RuleValue and WAValue are normalized, *i.e.* (i-min)/(max-min). The following shows weighting in different levels and explanation of formula:

$$WAValue(y_n(s)) = \sum_{level=1}^{6} \theta_{level} * WA_{level}(y_n(s)) \tag{3}$$

$$WA_{level}(y_n(s)) = \prod_{all\_word\_association\_for\_y_n(s)} P(Modify \mid Head) \tag{4}$$

After semantic probability collocating with rule probability, we hope to find the best tree $y^*(s)$.

$$y^*(s) = \arg\max Value(y_n(s)) \quad /^* Yi \text{ on all } i \tag{5}$$

We calculate related λ and θ values from the development sets. The development sets are adopted from trees in training data. In evaluation, we substitute λ and θ for every interval of 0.1 from 0 to 1. Then, we find out the best results in certain probability. The experiment results will be shown in the following section. Moreover, we justify whether the word associations are reasonable.

For instance, the following example has eight different ambiguous parsing results produced by the parser.

Input segmentation with PoS tag: 我們(Nh) 都(D) 喜歡(VK) 蝴蝶(Na)
Parsing results:
#1:1.[0] S(NP(Head:Nh:我們)|D:都|Head:VK:喜歡|NP(Head:Na:蝴蝶))#
#1:2.[0] NP(Nh:我們|Head:NP(VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶))#
#1:3.[0] VP(PP(Head:Nh:我們)|VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶)#
#1:4.[0] NP(VP(Head:Nh:我們)|Head:NP(VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶))#
#1:5.[0] VP(Head:VP(VP(Head:Nh:我們)|VP(D:都|Head:VK:喜歡))|NP(Head:Na:蝴蝶))#
#1:6.[0] NP(S(NP(Head:Nh:我們)|D:都|Head:VK:喜歡)|Head:Na:蝴蝶)#
#1:7.[0] VP(PP(Head:Nh:我們)|Head:VP(VP(D:都|Head:VK:喜歡)|VP(Head:Na:蝴蝶)))#
#1:8.[0] VP(Head:VP(VP(Head:Nh:我們)|VP(Head:D:都))|Head:VP(Head:VK:喜歡|NP(Head:Na:蝴蝶)))#

	Prob (log_2)
Rule	*-23.74*

Type	**WA**	**Prob (log_2)**
Level-1 (HWC_WC)	(我們/Nh_H/喜歡/VK) (都/D_H/喜歡/VK) (H/喜歡/VK_蝴碟/Na)	$log_2(76/21528)+log_2(578/21528)+$ $log_2(2/12200)$ = *-25.9395936826742*
Level-2 (HW_W)	(我們_H/喜歡) (都_H/喜歡) (H/喜歡_蝴碟)	$log_2(76/21528)+log_2(578/21528)+$ $log_2(2/12200)$ = *-25.9395936826742*
Level-3 (HC_WC)	(我們/Nh_H/VK) (都/D_H/VK) (H/VK_蝴碟/Na)	$log_2(25520/3235010)+log_2(49025/3235010)+$ $log_2(8/2501420)$ = *-31.2844226460991*
Level-4 (HW_C)	(Nh_H/喜歡) (D_H/喜歡) (H/喜歡_Na)	$log_2(3257/21528)+log_2(6160/21528)+$ $log_2(2927/11741)$ = -6.53387135079941
Level-5 (HC_C)	(Nh_H/VK) (D_H/VK) (H/VK_Na)	$log_2(230163/3235010)+log_2(1086580/3235010)+$ $log_2(575635/2601356)$ = *-7.56305573913316*
Level-6 (HS_S)	(我們_H/VK 喜) (D 都_H/VK 喜) (H/VK 喜_Na 碟)	$log_2(81/23809)+ log_2(586/23809)+$ $log_2(2/13986)$ = *-26.3155277463539*

Figure 4. An Example of Rule calculationand and WA probability.

Figure 4 shows the WA values of the first sentence at each level. Similarly the WA data are produced for all other input sentences. Then, we derive the evaluation values $Value(y_n(s))$ for each ambiguous sentence and find the best result with respect to different weights.

5. Experimental Results

The parsing performance and our evaluating model are evaluated by standard PARSEVAL metrics. In our experiments, we only use sentences longer than 6 words for our testing, since Hsieh *et al*. [2005] found that the bracketing *f*-score of short sentence (the length of a sentence is from 1 to 5 words) is over 90%. We use the *n*-best tree structures produced from "Rule type-3" mentioned in the Section 2. The oracle 50-best and the top 1-best bracketed f-scores of "Rule type-3" are listed in Table 4. Take the data of Sinica for example, we find that for the 50-best results, the oracle score is 90.11%. In contrast the 1-best *f*-score is 83.09%.

Table 4. The bracketed f-scores of 1-best and oracle performance of 50-best. (sentence length ≥ 6)

Top *n*-best	Testing data		
	Sinica	Sinorama	Textbook
1-best	83.09	77.54	83.19
50-best	90.11	87.44	89.94

To simplify our evaluation model, we try to find the most effective levels of associations first. In turn, the parser uses only one level of association and rule probabilities to select the best structure from *n* candidates. That is:

$$WAValue(y_n(s)) = WA_{level}(y_n(s)) = \prod_{all\_word\_association\_for\_y_n(s)} P(Modify \mid Head) \tag{6}$$

Figure 5 displays the bracketing *f*-scores of testing data for each different level of association. The best results of Level-1 slightly surpass that of Level-2; results of Level-6 overtake that of Level-3; Level-6 has better performance than Level-5. Therefore, in considering type of information, data coverage, and dimension reduction only three levels (Level-1, Level-4 and Level-6) are taken into consideration to form the final evaluation model.

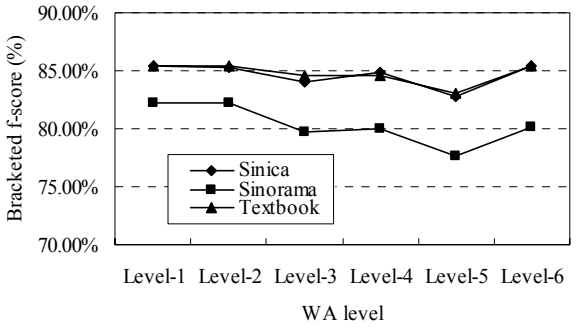


Figure 5. Matching rule with WA value in each level (sentence length ≥ 6).

Finally, we adjust the weights of L1, L4, and L6 associations and rule probabilities to evaluate the plausibility of structures from the 50-best parses tree of the developing data and the results of experiments on the three testing data are shown in Table 5. For our experiments, λ =0.7, θ_1=0.7, θ_4=0.3, and θ_6=0.5.

Table 5. The bracketed f-scores of 50-best parses (sentence length ≥ 6)

Models	Testing data		
	Sinica	Sinorama	Textbook
R, L1, L4, L6	86.59	82.81	85.97
1-best	83.09	77.54	83.19
50-best	90.11	87.44	89.94

From the results shown in Table 5, we see that semantic information is effective in finding better structures. About 3.5%~5.2% of the bracketing *f*-scores are raised. In Charniak *et al.* [2005], the *f*-score was improved from 89.7% (without re-ranking) to 91.02% (with re-ranking) for English[6]; the oracle *f*-score was 96.8% for *n*-best in their paper. We also believe that with more data parsed, better word-association values will be obtained; hence, the parsing performance will be improved by self-learning. Our WA was first extracted from the 1-best result from parser. With the parser producing *n*-best and the evaluating system finding the best structure, we can continuously derive more and better word associations. Similarly, if we have a better WA referent statistic, we should be able to choose the better structure. This is the idea of how self-learning works. The left side of Figure 6 denotes how we produce knowledge initially, and the right side of Figure 6 explains the repeated procedure of automatic knowledge extraction and accumulation. From the results shown in Table 4 and Table 5, we see that there is much space for improvement.

[6] The English parser has better evaluating results than the Chinese one due to the better performance of the parser and language differences. The charateristic of a strictly regulated grammar in English gives an advantage in parsing. Nonetheless, we have to admit that there is plenty of room for improvement in Chinese parsing.

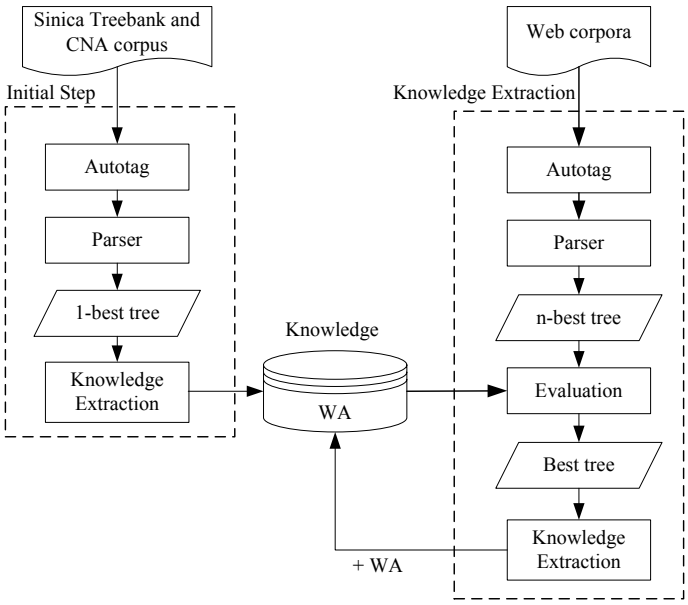


Figure 6. Procedure of self-learning.

6. Further Experiments on Sentences with Automatic PoS Tagging

Perfect testing data was used in the above experiments without considering PoS tagging errors. However, in reality, PoS tagging errors will degenerate parsing performance. The real parsing performance of accepting input from a PoS tagging system is shown in Table 6(1). In this table, "Autotag" means to markup the best PoS on the segmented data. The naïve approach to overcome the PoS tagging errors is to delay some of the ambiguous PoS resolution for words with lower confidence tagging scores and leave the ambiguous PoS to be resolved in the parsing stage. In Tsai *et al.* [2003], the tagging confidence of each word is measured by the following value:

$$\text{Confidence value} = \frac{P(c_1, w)}{P(c_1, w) + P(c_2, w)} \tag{7}$$

where P(c1,w) and P(c2,w) are probabilities assigned by the tagging model for the best candidate "c1,w" and the second best candidate "c2,w". Some examples follow:

confidence value=1.0
他({Nh,Nes}) 叫({VG,VF}) 李四(Nb) 撿({VC,VB}) 皮球(Na)
confidence value=0.8
他(Nh) 叫({VG,VF}) 李四(Nb) 撿(VC) 皮球(Na)
confidence value<0.5
他(Nh) 叫(VF) 李四(Nb) 撿(VC) 皮球(Na)

In Table 6(2), "Autotag with confidence value=1.0" means that if confidence value ≦ 1.0, we list all possible PoSs for the parser to decide. The experimental results of the 1-best, Table 6(2), show that delaying ambiguous PoS resolution does not improve parsing performance, since PoS ambiguities increase structural ambiguities and the PCFG parser is not robust enough to select better syntactic structures. However, for the experiment of 50-best, take the oracle score as the example; the 50-best oracle *f*-scores shown in Table 6(2) are better than the results without leaving ambiguous tags as shown in Table 6(1). Therefore, it is more likely to find better results after applying our evaluation model on the set of data with better oracle scores. Hence, we try to see the power of our evaluation model by leaving ambiguous PoS tags for the testing data.

Table 6. Oracle bracketed f-scores of different autotag for parsing: (1)Autotag; (2)Autotag with confidence value = 1.0.

Top *n*-best		Testing data		
		Sinica	Sinorama	Textbook
(1)	1-best	75.31	72.05	79.27
	50-best	84.09	83.36	87.54
(2)	1-best	73.41	68.34	77.83
	50-best	86.45	83.99	88.83

We then apply our evaluation model to select the best structure from the 50-best parses. The results are shown in Table 7. The experiment above takes "Rule type-3" for *n*-best parses. The bracketed *f*-score is raised from the original 73.41% to 79.34%, for about 4% improvement in the Sinica testing data. Sinorama data is improved from 68.34% to 74.78%. Textbook data is from 77.83% to 82.59%. This proves that our evaluating model is robust enough to handle ambiguous PoS tagging and produces better results than solely using the unique tag produced by Autotag.

Table 7. The bracketed f-scores in Autotag with confidence value=1.0 and 50-best parses (sentence length ≧ 6).

Models	Testing data		
	Sinica	Sinorama	Textbook
R, L1, L4, L6	79.34	74.78	82.59
1-best	73.41	68.34	77.83
50-best	86.45	83.99	88.83

7. Conclusion

Parsers of any language aim to correctly analyze the syntactic structure of a sentence, often with the help of semantic information. This paper shows a self-learning method to produce imperfect (due to errors produced by automatic parsing) but unlimited amount of word association data to evaluate the n-best trees produced by a feature-extended PCFG grammar. We prove that, although the statistical association strengths produced by automatic parsing are not perfect, the extracted data is reliable enough in measuring plausibility of ambiguous structures. The parser with this WA evaluation is considerably superior to those without evaluation. We believe that the above iterative learning processes can improve parsing performances automatically by learning word-dependence knowledge continuously from web. We also propose a method to modify our grammars to increase the oracle scores of the produced *n*-best sentences.

On the other hand, we offer a general syntactic and semantic evaluation model. We input *n*-best parses to our evaluating model. The evaluating model selects the best parse from this set of parses using a rule and semantic probability. The system we described, using the standard PARSEVAL framework, has a bracketed *f*-score of the selected trees, which is 86.59% higher than the original 1-best. Furthermore, the ambiguous PoS of a word is also parsed and evaluated on *n*-best, and we prove that our evaluating model is robust enough to improve parsing results on sentences with ambiguous PoS tagging.

From our experiment results, we find that sentences with coordinate structures are more difficult to deal with. The information of semantic parallelism instead of semantic dependencies is required for solving conjunctive structures. The extracted word associations don't have enough discriminative power to resolve both syntactic and semantic symmetry of conjunctive structures. The possible improvement may come from modifying the extraction method or predicting their plausible ranges before parsing. As to other difficult sentences, for example, in Figure 2, the coverage rate of Level 2 (HW_W) associations is only about 70%, which is far less than needed. We may expand our data to read more web texts to resolve this problem.

In future research, we plan to improve the quality of word-association. Four aspects need to be addressed: improving the accuracy of the PoS tagger, enhancing the parser's ability to solve common mistakes (such as parsing conjunctive structures), extracting more word associations by reading, and parsing text from web. As to the evaluation model, properly corresponding semantic classifications from coarse to fine-grained categories are needed in Level-6.

Acknowledgments

This research was supported in part by National Science Council under Grant NSC 95-2422-H-001-008- and National Digital Archives Program Grant 95-0210-29-戊-13-09-00-2.

Reference

Black, E., S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski, "A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars," In *Proceedings of the Workshop on Speech and Natural language*, 1991, pp. 306-311.

Charniak, E., and M. Johnson, "Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking," In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 2005, Ann Arbor, MI, pp. 173-180.

Chen, K.-J., C.-R. Huang, F.-Y. Chen, C.-C. Luo, M.-C. Chang, C.-J. Chen, and Z.-M. Gao, "Sinica Treebank: design criteria, representational issues and implementation," In *Anne Abeille, (ed.): Building and Using Parsed Corpora. Text, Speech and Language Technology*, 2003, 20, pp. 231-248.

Chen, Y., M. Asahara, and Y. Matsumoto, "Deterministic Dependency Structure Analyzer for Chinese," In *Proceedings of the First International Join Conference on Natural Language Processing*, 2004, Sanya City, Hainan Island, China, pp. 135-140.

Chiu, C.-M., J.-Q. Luo, and K.-J. Chen, "Compositional semantics of mandarin affix verbs." In *Proceedings of ROCLING XVI: Conference on Computational Linguistics and Speech Processing*, 2004, Taipei, pp. 131-139.

CKIP (Chinese Knowledge Information processing), "The categorical analysis of Chinese," Technical Report No. 93-05, Institute of Information Science Academia Sinica, Taipei, 1993.

Collins, M., "Head-driven statistical models for natural language parsing," *PhD thesis*, University of Pennsylvania, 1999.

Collins, M., "Discriminative reranking for natural language parsing," In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, 2000, Morgan Kaufmann, San Francisco, CA, pp. 175-182.

Hsieh, Y.-M., D.-C. Yang, and K.-J. Chen, "Linguistically-motivated grammar extraction, generalization and adaptation," In *Proceedings of the Second International Join Conference on Natural Language Processing*, 2005, Jeju Island, Republic of Korea, pp. 177-187.

Johnson, M., "PCFG models of linguistic tree representations," *Computational Linguistics*, 1998, 24(4), pp. 613-632.

Klein, D., and C. D. Manning, "Accurate unlexicalized parsing," *In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003, Sapporo, Japan, pp. 423-430.

Tsai, Y.-F., and K.-J. Chen, "Context-rule model for PoS tagging," In *Proceedings of 17th Pacific Asia Conference on Language, Information and Computation (PACLIC 17)*, 2003, COLIPS, Sentosa, Singapore, pp. 146-151.

Xiong, D., S. Li, Q. Liu, S. Lin, and Y. Qian, "Parsing the Penn Chinese Treebank with semantic knowledge," In *Proceedings of the Second International Join Conference on Natural Language Processing*, 2005, Jeju Island, Republic of Korea, pp. 70-81.

Appendix A. Transformation algorithm

Notation:

WORD: user input Word
POS: user input PoS of the word
CLASS: transformation class of the word
Affix(WORD): input WORD to find mapping affix from table
Prefix(WORD): prefix of the WORD
Suffix(WORD): suffix of the WORD
DM(WORD): input Word to find DM category

Input: *WORD, POS*

Output: *CLASS*

Initial Step:

CLASS=WORD;
if *WORD* in affix table then *CLASS=affix(WORD)*;
if *POS* is verb or adverb then *CLASS=POS+prefix(WORD)*;
if *POS* is noun then *CLASS=POS+suffix(WORD)*;

Mapping Step:

if *POS* is non-predicative adjective then *CLASS='A'+prefix(WORD)*; /* e.g. A */
if *POS* is preposition then *CLASS='P'+suffix(WORD)*; /* e.g. P */
if *POS* is SHI then *CLASS='SHI'*; /* e.g. 是 */
if *POS* is V_2 then *CLASS='V_2'*; /* e.g. 有 */
if *POS* is DM or Measure and exist in DM table then *CLASS=DM(WORD)*;
/* e.g. DM/Nf */
if *POS* is conjunction then *CLASS=POS+prefix(WORD)*; /* e.g. Caa/Cab/Cba/Cbb */
if *POS* is determinative then *CLASS=POS*; /* e.g. Nep/Neqa/Neqb/Nes/Neu */
if *POS* is pronoun then *CLASS=WORD*; /* e.g. Nh */
if *POS* is time noun then *CLASS='Time'*; /* e.g. Nd */
if *POS* is Postposition/Place Noun/Localizer then *CLASS='Location'*;
/* e.g. Ng/Nc/Ncd */
if *POS* is Proper Noun and is family names then *CLASS='PersonalName'*; /* e.g. Nb */
if *POS* is aspectual adverb then *CLASS=POS* /* e.g. Di */
if *POS* is pre/post-verbal adverb of degree then *CLASS='Df'+suffix(Word)*
/*e.g. Dfa/Dfb */
if *POS* is VD/VCL/VL then *CLASS=POS+suffix(WORD)*

Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 217-238

A Comparative Study of Histogram Equalization (HEQ) for Robust Speech Recognition

Shih-Hsiang Lin*, Yao-Ming Yeh*, and Berlin Chen*

Abstract

The performance of current automatic speech recognition (ASR) systems often deteriorates radically when the input speech is corrupted by various kinds of noise sources. Quite a few techniques have been proposed to improve ASR robustness over the past several years. Histogram equalization (HEQ) is one of the most efficient techniques that have been used to reduce the mismatch between training and test acoustic conditions. This paper presents a comparative study of various HEQ approaches for robust ASR. Two representative HEQ approaches, namely, the table-based histogram equalization (THEQ) and the quantile-based histogram equalization (QHEQ), were first investigated. Then, a polynomial-fit histogram equalization (PHEQ) approach, exploring the use of the data fitting scheme to efficiently approximate the inverse of the cumulative density function of training speech for HEQ, was proposed. Moreover, the temporal average (TA) operation was also performed on the feature vector components to alleviate the influence of sharp peaks and valleys caused by non-stationary noises. All the experiments were carried out on the Aurora 2 database and task. Very encouraging results were initially demonstrated. The best recognition performance was achieved by combing PHEQ with TA. Relative word error rate reductions of 68% and 40% over the MFCC-based baseline system, respectively, for clean- and multi- condition training, were obtained.

Keywords: Automatic Speech Recognition, Robustness, Histogram Equalization, Data Fitting, Temporal Average

1. INTRODUCTION

With the successful development of much smaller electronic devices and the popularity of wireless communication and networking, it is widely believed that speech will play a more

* Department of Computer Science & Information Engineering, National Taiwan Normal University, Taipei, Taiwan
E-mail: { shlin, ymyeh, berlin }@csie.ntnu.edu.tw

[Received March 2, 2007; Revised July 4, 2007; Accepted July 12, 2007]

active role and will serve as the major human machine interface (HMI) for the interaction between people and different kinds of smart devices in the near future [Lee and Chen 2005]. Therefore, automatic speech recognition (ASR) has long been one of the major preoccupations of research in the speech and language processing community. Nevertheless, varying environmental effects, such as ambient noise, noises caused by the recording equipment and transmission channels, etc., often lead to a severe mismatch between the acoustic conditions for training and test. Such a mismatch will no doubt cause substantial degradation in the performance of an ASR system. Substantial effort has been made and a large number of techniques have been presented in the last few decades to cope with this issue for improving ASR performance [Gong 1995; Junqua *et al*. 1996; Huang *et al*. 2001]. In general, they fall into three main categories [Gong 1995]:

- Speech enhancement, which removes the noise from the observed speech signal.
- Robust speech features extraction, which searches for noise resistant and robust features.
- Acoustic model adaptation, which transforms acoustic models from the training (clean) space to the test (noisy) space.

Techniques of each of the above three categories have their own reasons for superiority and their own limitations. In practical implementation, acoustic model adaptation often yields the best recognition performance, because it directly adjusts the acoustic models parameters (*e.g*., the mean vectors or covariance matrices of mixture Gaussian models) to accommodate the uncertainty caused by noisy environments. Representative techniques, include, but are not limited to, the maximum a posteriori (MAP) adaptation [Gauvain and Lee 1994; Huo *et al*. 1995], the maximum likelihood linear regression (MLLR) [Leggeter and Woodland 1995; Gales 1998], etc. However, such techniques generally require a sufficient amount of extra adaptation data (either with or without reference transcripts) and a significant computational cost in comparison with the other two categories. Moreover, most of the speech enhancement techniques target enhancing the signal-to-noise ratio (SNR) but not necessarily at improving the speech recognition accuracy. On the other hand, robust speech feature extraction techniques can be further divided into two subcategories, *i.e*., model-based compensation and feature space normalization. Model-based compensation assumes the mismatch between clean and noisy acoustic conditions can be modeled by a stochastic process. The associated compensation models can be estimated in the training phase, and then exploited to restore the feature vectors in the test phase. Typical techniques of this subcategory, include, but are not limited to, the minimum mean square error log spectral amplitude estimator (MMSE-LSA) [Ephraim and Malah 1985], the vector Taylor series (VTS) [Moreno 1996], the stochastic vector mapping (SVM) [Wu and Huo 2006], the multi-environment model-based linear normalization (MEMLIN) [Buera *et al*. 2007], etc.

Feature space normalization is believed to be a simpler and more effective way to compensate for the mismatch caused by noise, and it has also demonstrated the capability to prevent the degradation of ASR performance under various noisy environments. Several attractive techniques have been successfully developed and integrated into the state-of-the-art ASR systems. As an example, the cepstral mean subtraction (CMS) [Furui 1981] is a simple but effective technique for removing the time-invariant distortion introduced by the transmission channel; while a natural extension of CMS, called the cepstral mean and variance normalization (CMVN) [Vikki and Laurila 1998], attempts to normalize not only the means of speech features but also their variances. Although these two techniques have already shown their capabilities in compensating for channel distortions and some side effects resulting from additive noises, their linear properties still make them inadequate in tackling the nonlinear distortions caused by various noisy environments [Torre *et al*. 2005]. Accordingly, a considerable amount of work on seeking more general solutions for feature space normalization has been done over the past several years. For example, not content with using either CMN or CMVN merely to normalize the first or the first two moments of the probability distributions of speech features, some researchers have extended the principal idea of CMN and CMVN to the normalization of the third [Suk *et al*. 1999] or even more higher order moments of the probability distributions of speech features [Hsu and Lee 2004, 2006]. On the other hand, the histogram equalization (HEQ) techniques also have gained much attention, and have been widely investigated in recent years [Dharanipragada and Padmanabhan 2000; Molau *et al*. 2005; Torre *et al*. 2005; Hilger and Ney 2006; Lin *et al*. 2006]. HEQ seeks for a transformation mechanism that can map the distribution of the test speech onto a predefined (or reference) distribution utilizing the relationship between the cumulative distribution functions (CDFs) of the test speech and those of the training (or reference) speech. Therefore, HEQ not only attempts to match the means and variances of speech features but also completely match the distributions of speech features between training and test. More specifically, HEQ normalizes all moments of the probability distributions of test speech features to those of the reference ones. However, most of the current HEQ techniques still have some inherent drawbacks for practical usage. For example, they require either large storage consumption or considerable online computational overhead, which might make them infeasible when being applied to the ASR systems built on devices with limited resources, such as personal digital assistants (PDAs), smart phones and embedded systems, etc.

With these observations in mind, in this paper we present a comparative study of various HEQ approaches for robust speech recognition. Two representative HEQ approaches, namely, the table-based histogram equalization (THEQ) and the quantile-based histogram equalization (QHEQ), were first investigated. Then, a polynomial-fit histogram equalization (PHEQ)

approach, exploring the use of the data fitting scheme to efficiently approximate the inverse of the cumulative density function of training speech for HEQ, was proposed. Moreover, the temporal average (TA) operation was also performed on the feature vector components to alleviate the influence of sharp peaks and valleys that were caused by non-stationary noises.

The remainder of this paper is organized as follows. Section 2 describes the basic concept of HEQ and reviews two representative HEQ approaches, namely, THEQ and QHEQ. Section 3 elucidates our proposed HEQ approach, namely, PHEQ, and also briefly introduces several standard temporal average operations. Section 4 gives an overview of the Aurora 2 database as well as a description of the experimental setup, while the corresponding experimental results and discussions are also presented in this section. Finally, conclusions are drawn in Section 5.

2. HISTOGRAM EQUALIZATION (HEQ)

2.1 Theoretical Foundation of HEQ

Histogram equalization is a popular feature compensation technique that has been well studied and practiced in the field of image processing for normalizing the visual features of digital images, such as the brightness, grey-level scale, contrast, and so forth. It has also been introduced to the field of speech processing for normalizing the speech features for robust ASR, and many good approaches have been continuously proposed and reported in the literature [Dharanipragada and Padmanabhan 2000; Molau *et al*. 2003; Torre *et al*. 2005; Hilger and Ney 2006; Lin *et al*. 2006]. Meanwhile, HEQ has shown its superiority over the conventional linear normalization techniques, such as CMN and CMVN, for robust ASR. One additional advantage of HEQ is that it can be easily incorporated with most feature representations and other robustness techniques without the need of any prior knowledge of the actual distortions caused by different kinds of noises.

Theoretically, HEQ has its roots in the assumptions that the transformed speech feature distributions of the test (or noisy) data should be identical to that of the training (or reference) data and each feature vector dimension can be normalized independently of each other. The speech feature vectors can be estimated either from the Mel-frequency filter bank outputs [Molau 2003; Hilger and Ney 2006] or from the cepstral coefficients [Segura *et al*. 2004; Torre *et al*. 2005; Lin *et al*. 2006]. Since each feature vector dimension is considered independently, from now on, the dimension index of each feature vector component will be omitted from the discussion for the simplicity of notation unless otherwise stated. Under the above two assumptions, the aim of HEQ is to find a transformation that can convert the distribution of each feature vector component of the input (or test) speech into a predefined target distribution which corresponds to that of the training (or reference) speech. The basic

idea of HEQ is illustrated in Figure 1.

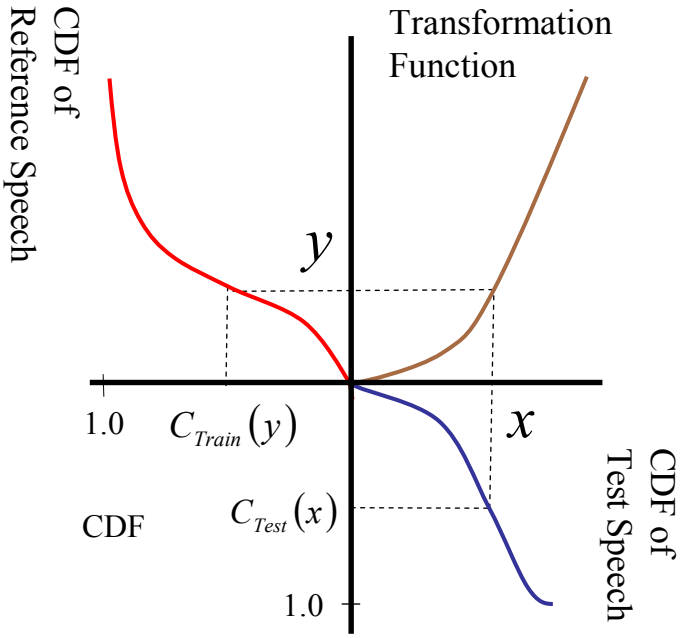


Figure 1. The basic idea of HEQ.

Accordingly, HEQ attempts not only to match the means and variances of the speech features, but also to completely match the speech feature distributions of training and test data. Phrased another way, HEQ normalizes all the moments of the probability distributions of the speech features. The formulation of HEQ is described as follows [Torre *et al*. 2005]. For each feature space dimension, let x be the feature vector component that follows the distribution $p_{Test}(x)$. A transformation function $F(x)$ converts x to y and follows a reference distribution $p_{Train}(y)$, according to the following expression:

$$p_{Train}(y) = p_{Test}(x)\frac{dx}{dy} = p_{Test}\left(F^{-1}(y)\right)\frac{dF^{-1}(y)}{dy}, \tag{1}$$

where $F^{-1}(y)$ is the inverse function of $F(x)$. Moreover, the relationship between the cumulative probability density functions (CDFs) associated with the test and training speech, respectively, is governed by:

$$\begin{aligned} C_{Test}(x) &= \int_{-\infty}^{x} p_{Test}(x')dx' \\ &= \int_{-\infty}^{F(x)} p_{Test}(F^{-1}(y'))\frac{dF^{-1}(y')}{dy'}dy' \\ &= \int_{-\infty}^{y} p_{Train}(y')dy'\Big|_{y=F(x)} \\ &= C_{Train}(y), \end{aligned} \tag{2}$$

where $C_{Test}(x)$ and $C_{Train}(y)$ are the CDFs for the test and training speech, respectively; y' is the corresponding output of the transformation function $F(x')$; and the transformation function $F(x)$ has the following property:

$$F(x) = C_{Train}^{-1}\left(C_{Test}(x)\right), \tag{3}$$

where C_{Train}^{-1} is the inverse function of C_{Train}.

It is worth noting that the reliability of CDF estimation will have a significant influence on the performance of HEQ. Due to the finite number of speech features being considered, the CDFs of speech features are usually approximated by the cumulative histograms of speech features for practical implementation. The CDFs of speech features can be accurately and reliably approximated when there is a large amount of data available. On the contrary, such approximation will probably not be accurate enough when the (test) speech utterance becomes much shorter. Several studies have shown that the order-statistics based method tends to be more accurate than the cumulative-histogram based when the amount of speech data is insufficient for reliable approximation of CDFs [Segura *et al*. 2004; Torre *et al*. 2005].

2.2 Table-Based Histogram Equalization (THEQ)

The table-based histogram equalization (THEQ) was first proposed by Dharanipragada and Padmanabhan [Dharanipragada and Padmanabhan 2000] and is a non-parametric method to let the distributions of the test speech match those of the training speech. THEQ uses a cumulative histogram to estimate the corresponding CDF value of each feature vector component y. During the training phase, the cumulative histogram of each feature vector component y of the training data is constructed as follows. The range of values of each feature vector dimension over the entire training data is first determined by finding the feature vector components $y_{\max}$ and $y_{\min}$ that have the maximum and minimum values, respectively. Let K be the total number of histogram bins and the range $\left[y_{\min}, y_{\max}\right]$ is then divided into K non-overlapped bins of equal size, $\{B_0, B_1, \cdots B_{K-1}\}$. Next, the entire training data is scanned once and each individual feature vector component falls exactly into one bin. Thus, if we let N be the total number of training feature vector components of one specific dimension and n_i be the number of feature vector components of that dimension belonging to B_i, the probability of feature vector components of that dimension being in B_i is approximated by:

$$p_{Train}(B_i) = \frac{n_i}{N}. \tag{4}$$

The mean $\overline{y}_{B_i}$ of each bin i is taken as one of the representative outputs of the transformation function $F(x)$ and the approximate CDF value $C_{Train}(y)$ of the feature vector component y that belongs to B_i is calculated by:

$$C_{Train}(y) = \sum_{j=0}^{i} p_{Train}(B_j). \tag{5}$$

Finally, a look-up table consisting of all possible distinct reference pairs $\left(C_{Train}(y), \overline{y}_{B_i}\right)$ is constructed, where $C_{Train}(y)$ is taken as the key and $\overline{y}_{B_i}$ is the corresponding restored value. During the test phase, the CDF estimation of the test utterance can be done in the same way by using the cumulative histograms of itself. The restored value of each feature vector component x of the test utterance is obtained by taken its approximate CDF value $C_{Test}(x)$ as the key to finding the corresponding transformed (restored) value in the look-up table.

However, the normalization of the test data alone results in only a moderate gain of performance improvement. It has been suggested that one should normalize the training data in the same way to achieve good performance [Molau *et al*. 2003]. On the other hand, because a set of cumulative histograms of all speech feature vector dimensions of the training data has to be kept in memory for the table-lookup of restored feature values, THEQ needs large disk storage consumption and its associated table-lookup procedure is also time-consuming, which might make THEQ not very feasible for ASR systems that are built into devices with limited resources, such as PDAs, smart phones and embedded systems, etc.

2.3 Quantile-Based Histogram Equalization (QHEQ)

The quantile-based histogram equalization (QHEQ) is a parametric type of histogram equalization. QHEQ attempts to calibrate the CDF of each feature vector component of the test speech to that of the training speech in a quantile-corrective manner instead of a full-match of the cumulative histogram as done by THEQ, described earlier in Section 2.2. Normally, QHEQ only needs a small number of quantiles (usually the number is set to 4) for reliable estimation [Hilger and Ney 2001, 2006]. A transformation function $H(x)$ is calculated by minimizing the mismatch between the quantiles of the test utterance and those of the training data. The transformation function $H(x)$ is a power function applied to each feature vector component x, which attempts to make the CDF of the equalized feature vector component match that observed in training. Before the actual application of the transformation function $H(x)$, each feature vector component x is first scaled down into the interval $[0,1]$ by being divided by the maximum value Q_K over the entire utterance. Then, the transformation function $H(x)$ is applied to x and the transformed (or restored) value of x is scaled back to the original value range [Hilger and Ney 2006]:

$$H(x) = Q_K \left(\alpha \left(\frac{x}{Q_K} \right)^{\gamma} + (1-\alpha) \frac{x}{Q_K} \right), \tag{6}$$

where K is the total number of quantiles; Q_K is the maximum value over the entire utterance; and α and γ are the transformation parameters. For each feature vector dimension, α and γ are chosen to minimize the squared distance between the quantiles $H(Q_k)$ of the test utterance and the quantiles Q_k^{Train} of the training data by using the following equation:

$$\{\alpha, \gamma\} = \arg\min_{\{\alpha,\gamma\}} \left(\sum_{k=1}^{K-1} \left(H(Q_k) - Q_k^{Train} \right)^2 \right). \tag{7}$$

In summary, QHEQ allows the estimation of the transformation function $H(x)$ to merely rely on a single test utterance (or extremely, a very short utterance), without the need of an additional set of adaptation data [Hilger and Ney 2006]. However, in order to find the optimum transformation parameters for each feature vector dimension, an exhaustive online grid search is required, which, in fact, is very time-consuming.

3. IMPROVED APPROACHES

3.1 Polynomial-Fit Histogram Equalization (PHEQ)

In contrast to the above table-lookup or quantile based approaches, we propose a polynomial-fit histogram equalization (PHEQ) approach which explores the use of the data fitting scheme to efficiently approximate the inverse functions of the CDFs of the training speech for HEQ [Lin *et al*. 2006]. Data fitting is a mathematical optimization method which, when given a series of data points (u_i, v_i) with $i = 1, \cdots, N$, attempts to find a function $G(u_i)$ whose output $\tilde{v}_i$ closely approximates v_i. That is, it minimizes the sum of the squares error (or the squares of the ordinate differences) between the points $(u_i, \tilde{v}_i)$ and their corresponding points (u_i, v_i) in the data. The function $G(u_i)$ to be estimated can be either linear or nonlinear in its coefficients. For example, if $G(u_i)$ is a linear M-order polynomial function:

$$G(u_i) = \tilde{v}_i = a_0 + a_1 u_i + a_2 {u_i}^2 + \cdots + a_M {u_i}^M, \tag{8}$$

where $a_0, a_1, \cdots, a_M$ are the coefficients, then its corresponding squares error can be defined by

$$E^2 = \sum_{i=1}^{N} (v_i - \tilde{v}_i)^2 = \sum_{i=1}^{N} \left(v_i - \sum_{m=0}^{M} a_m {u_i}^m \right)^2. \tag{9}$$

PHEQ makes use of such data fitting (or so-called least squares regression) scheme to estimate the inverse functions of the CDFs of the training speech. For each speech feature vector dimension of the training data, given the pair of the CDF value $C_{Train}(y_i)$ of the vector component y_i and y_i itself, the linear polynomial function $G\left(C_{Train}(y_i)\right)$ with output $\tilde{y}_i$ can be expressed as:

$$G\left(C_{Train}(y_i)\right) = \tilde{y}_i = \sum_{m=0}^{M} a_m \left(C_{Train}(y_i)\right)^m, \tag{10}$$

where the coefficients a_m can be estimated by minimizing the squares error expressed in the following equation:

$$E'^2 = \sum_{i=1}^{N} \left(y_i - \tilde{y}_i\right)^2 = \sum_{i=1}^{N} \left(y_i - \sum_{m=0}^{M} a_m \left(C_{Train}(y_i)\right)^m \right)^2, \tag{11}$$

where N is the total number of training speech feature vectors. In implementation, we used the order-statistics based method instead of the cumulative-histogram based method to obtain the approximate CDF values. For the feature vector component sequence $Y = [y_1, \cdots, y_i, \cdots, y_N]$ of a specific dimension of a speech utterance, the corresponding CDF value of each feature component y_i can be approximated by the following two steps:

Step1: The sequence $Y = [y_1, \cdots, y_i, \cdots, y_N]$ is first sorted according to the values of the feature vector components in ascending order.

Step2: The order-statistics based approximation of the CDF value of a feature vector component y_i is then given as:

$$C(y_i) \approx \frac{S_{pos}(y_i) - 0.5}{N} \tag{12}$$

where $S_{pos}(y_i)$ is a function that returns the rank of y_i in ascending order of the values of the feature vector components of the sequence $Y = [y_1, \cdots, y_i, \cdots, y_N]$. Therefore, for each utterance, Equation (12) can be used to approximate the CDF values of the feature vector components of all dimensions. During the training phase, the polynomial functions of all dimensions are obtained by minimizing the squares error expressed in Equation (11). During the test phase, for each feature vector dimension, the feature vector components of the test utterance are simply sorted in ascending order of their values to obtain the approximate CDF values, which can be then taken as the inputs to the inverse function to obtain the corresponding restored component values.

The reason we choose the polynomial function here as the inverse function is mainly because it has a simple form, without the need of a complicated computational procedure, and

has moderate flexibility in controlling the shape of the function. Though the polynomial function is efficient in delineating the transformation function, it is worth mentioning that the polynomial function to some extent has its inherent limitations. For example, high order polynomial functions might lead to over-fitting of the training data. Moreover, the polynomial function provides good fits for input data points that are located within the range of values of the training data, but would also probably have rapid deterioration when the input data points are located outside the range of values of the training data.

3.2 Temporal Average (TA)

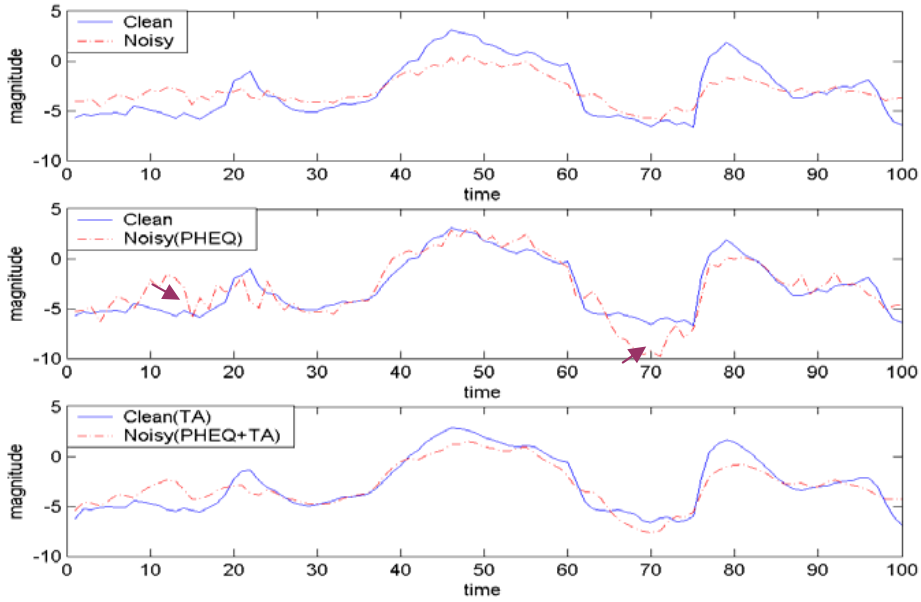


Figure 2. The 2th cepstral feature component sequence of an utterance

Though the above HEQ approaches are very effective in matching the global feature statistics of the test (or noisy) speech to that of the training (or reference) set, we found that some undesired sharp peaks or valleys of the feature vector component sequence caused by the non-stationary noises often occurring during the equalization process. This phenomenon is illustrated in the upper and middle parts of Figure 2. Therefore, we believe that a rigorous smoothing operation further performed on the time trajectory of the HEQ restored feature vector component sequence will be helpful for suppressing the extraordinary changes of component values. From the other perspective, temporal average can be treated as a low-pass filter. The basic idea of TA is quite similar to RelAtive SpecTrA (RASTA) [Hermansky and Morgan 1994] which aims to filter out the slow-varying or fast-varying artifacts (or noises) based on the evidence of human auditory perception. The main differences between TA and RASTA are the target (or feature domain) where the smoothing operation is performed and the

design of the temporal filters. The smoothing (or temporal average) operation can be defined as one of the following forms [Chen and Bilmes 2007]:

- Non-Causal Moving Average

$$\hat{y}_t = \begin{cases} \dfrac{\sum_{i=-L}^{L} \tilde{y}_{t+i}}{2L+1} & if\ L < t \le T-L, \\ \tilde{y}_t & otherwise \end{cases} \tag{13}$$

- Causal Moving Average

$$\hat{y}_t = \begin{cases} \dfrac{\sum_{i=0}^{L} \tilde{y}_{t-i}}{L+1} & if\ L < t \le T, \\ \tilde{y}_t & otherwise \end{cases} \tag{14}$$

- Non-Causal Auto Regression Moving Average

$$\hat{y}_t = \begin{cases} \dfrac{\sum_{i=1}^{L} \hat{y}_{t-i} + \sum_{j=0}^{L} \tilde{y}_{t+j}}{2L+1} & if\ L < t \le T-L, \\ \tilde{y}_t & otherwise \end{cases} \tag{15}$$

- Causal Auto Regression Moving Average

$$\hat{y}_t = \begin{cases} \dfrac{\sum_{i=1}^{L} \hat{y}_{t-i} + \sum_{j=0}^{L} \tilde{y}_{t-j}}{2L+1} & if\ L < t \le T, \\ \tilde{y}_t & otherwise \end{cases} \tag{16}$$

where $\tilde{y}_t$ denotes the HEQ restored feature vector component at speech frame t; L is the span order of temporal average operation; and $\hat{y}_t$ is the corresponding one after the temporal average operation. The feature vector component sequence obtained by Equation (13) is also shown in the lower part of Figure 2.

4. EXPERIEMENTAL RESULTS

4.1 Experimental Setup

The speech recognition experiments were conducted under various noise conditions using the Aurora-2 database and task [Hirsch and Pearce 2002]. The Aurora-2 database is a subset of the TI-DIGITS, which contains a set of connected digit utterances spoken in English; while the task consists of the recognition of the connected digit utterances interfered with various noise sources at different signal-to-noise ratios (SNRs), in which Test Sets A and B are artificially contaminated with eight different types of real-world noises (*e.g.*, subway noise, street noise,

babble noise, etc.) in a wide range of SNRs (-5 dB, 0 dB, 5 dB, 10 dB, 15 dB, 20 dB and Clean) and Test Set C additionally includes channel distortions. For the baseline system, the training and recognition tests used the HTK recognition toolkit [Young *et al*. 2005], following the original setup defined for the ETSI AURORA evaluations [Hirsch and Pearce 2002].

More specifically, each digit was modeled as a left-to-right continuous density hidden Markov model (CDHMM) with 16 states and three diagonal Gaussian mixtures per state. Two additional CDHMMs were defined for the silence. The first one had three states with six diagonal Gaussian mixtures per state for modeling the silence at the beginning and at the end of each utterance. The other one had one state with 6 diagonal Gaussian mixtures for modeling the inter-word short pause. In the front-end speech analysis, the frame length is 25 ms and the corresponding frame shift is 10 ms. Speech frames are pre-emphasized using a factor of 0.97, and the Hamming window is then applied. From a set of 23 Mel-scaled log filter banks outputs a 39-dimensional feature vector, consisting of 12 Mel-frequency cepstral coefficients (MFCCs), the 0-th cepstral coefficient, and the corresponding delta and acceleration coefficients, is extracted at each speech frame. The average word error rate (WER) results obtained by the MFCC-based baseline system are 45.44% and 14.65%, respectively, for clean- and multi-condition training, each of which is an average of the WER results of the test utterances respectively contaminated with eight types of noises under different SNR levels (0 dB to 20 dB) for the three sets (Sets A, B and C).

4.2 Experiments on HEQ Approached

Table 1. Average WER results (%) of THEQ for clean-condition training, with respect to different numbers of histogram bins and different sizes of table.

		Table Size							
		10	50	100	500	1000	5000	10000	50000
Histogram Bin Number	100	41.32	45.65	46.39	44.59	44.55	44.65	44.67	44.65
	500	33.21	28.60	25.44	22.42	22.42	22.41	22.45	22.41
	1000	29.63	24.19	22.12	19.19	19.04	19.46	19.88	19.87
	5000	28.13	23.72	20.68	18.22	18.02	18.18	18.19	18.10
	10000	27.64	23.50	20.50	18.33	18.10	18.13	18.30	18.32
	50000	27.46	23.30	20.29	18.58	18.41	18.46	18.47	18.45
	Order-Statistics	27.26	23.30	20.65	18.62	18.32	18.51	18.53	18.58

Table 2. Average WER results (%) of THEQ for multi-condition training, with respect to different numbers of histogram bins and different sizes of table.

		Table Size							
		10	50	100	500	1000	5000	10000	50000
Histogram Bin Number	100	19.46	22.27	23.81	23.85	23.96	24.05	24.06	24.07
	500	18.54	20.71	19.06	14.94	14.58	14.57	14.52	14.59
	1000	18.94	19.46	17.04	13.63	13.30	13.36	13.35	13.33
	5000	19.24	18.98	15.91	12.52	12.30	12.31	12.29	12.27
	10000	19.27	18.79	15.75	12.26	12.26	12.23	12.22	12.23
	50000	19.42	18.79	15.69	12.76	12.14	12.16	12.15	12.16
	Order-Statistics	19.43	18.91	15.73	12.79	12.18	12.17	12.17	12.17

In the first set of experiments, we compare the recognition performance when different numbers of the histogram bins and different sizes of the look-up table are applied for THEQ. Notice that the equalization was conducted on all dimensions of the feature vectors for the training and test data, and the approximation of the CDFs of the test speech was conducted in an utterance-by-utterance manner. The results are summarized in Tables 1 and 2 for clean- and multi-condition training, respectively. As can been seen, the recognition performance is very sensitive to the number of the histogram bins and the size of the look-up table. The WER is improved when either the number of the histogram bins or the size of the look-up table is increased. As compared to the MFCC-based baseline system, the best results of HEQ yield about 60% and 16% relative WER improvements for clean- and multi-condition training, respectively. These results suggest that a larger histogram bin number or table size can improve the recognition performance, however, at the cost of huge consumption of the memory storage. Moreover, THEQ is also time-consuming, because a huge set of cumulative histograms of all speech feature vector dimensions of the training data have to be kept in memory for the table-lookup of restored feature values. Furthermore, the CDF value of a feature vector component approximated by the cumulative-histogram based method is equivalent to that done by the order-statistics based method when the number of histogram bins is taken to be infinite.

In the next set of experiments, we investigate the use of different quantile numbers for QHEQ to see if the quantile number has any apparent effect on the recognition performance. The corresponding average WER results are shown in Table 3. As indicated by the results, it can be found the recognition performance is closely dependent on the quantile number. The transformation function $H(x)$ would tend to be too coarse to model the relationship between the test utterance and the training data when only few quantiles are being considered. On the contrary, the use of too many quantiles for the estimation of the transformation function

Table 3. Average WER results (%) of QHEQ, with respect to different quantile numbers.

	Quantile Number						
	2	3	4	5	8	16	32
Clean-Condition Training	24.02	23.67	22.86	23.00	24.93	24.83	24.95
Multi-Condition Training	11.63	11.25	10.23	10.24	12.36	12.32	12.36

Table 4. Average WER results (%) of PHEQ, with respect to different orders of the polynomial transformation functions.

	Polynomial Order						
	1-th	3-th	5-th	7-th	9-th	11-th	13-th
Clean-Condition Training	18.54	17.1	16.05	15.71	15.72	15.72	16.68
Multi-Condition Training	12.17	9.44	9.26	9.50	9.45	9.46	11.45

$H(x)$ might instead degrade the recognition performance [Hilger and Ney 2001]. However, the optimum number of quantiles is found to be four for the Aurora 2 task studied here, and the corresponding relative WER improvements over the MFCC-based baseline system are 50% and 30% for clean- and multi-condition training, respectively.

In the third set of experiments, we evaluate the performance of PHEQ with respect to different polynomial orders and the associated results are presented in Table 4. Due to the end behavior property of polynomial functions, even order polynomials are either "up" on both ends or "down" on both ends which is not appropriate to characterize the behavior of a cumulative distribution [Lial *et al*. 2006]. Therefore, only odd-order polynomials are utilized in this paper for PHEQ. As evidenced by the results shown in Table 4, the average WER results of PHEQ are slightly improved when the order of the polynomial function becomes higher. However, as the order increases, the polynomial function might sometimes tend to over-fit of the training data. The improvement of PHEQ seems to saturate when the order is set to seven. As is indicated, PHEQ yields about a relative WER improvement of 65% for clean-condition training, and 35% for multi-conditions training, as compared to the MFCC-based baseline system.

To go a step further, the average WER results under different SNR levels for the MFCC baseline, THEQ, QHEQ and PHEQ are shown in Tables 5 and 6, for clean- and multi-condition training, respectively. In the case of clean-condition training, these three HEQ approaches all yield significant improvement over the MFCC-based baseline, especially when the SNR level becomes much lower (*e.g.*, 10 dB, 5 dB or 0 dB). The average WERs for

Table 5. Average WER results (%) of the MFCC-based baseline system, THEQ, QHEQ and PHEQ for clean-condition training, with respect to different SNR levels.

	SNR Level						
	Clean	20 dB	15 dB	10 dB	5 dB	0 dB	-5 dB
MFCC	0.89	7.55	20.41	43.17	70.80	90.21	96.37
THEQ	1.73	3.61	5.69	10.22	21.66	47.41	77.91
QHEQ	0.82	2.05	4.14	10.84	30.90	66.11	86.72
PHEQ	0.92	1.83	3.45	7.52	18.84	45.78	76.77

Table 6. Average WER results (%) of the MFCC-based baseline system, THEQ, QHEQ and PHEQ for multi-condition training, with respect to different SNR levels.

	SNR Level						
	Clean	20 dB	15 dB	10 dB	5 dB	0 dB	-5 dB
MFCC	1.15	2.16	3.22	5.97	15.45	44.06	79.24
THEQ	1.10	2.24	3.53	6.52	15.63	40.60	73.39
QHEQ	2.15	2.02	2.74	5.10	10.32	29.46	57.96
PHEQ	1.34	1.65	2.43	4.19	10.14	27.96	62.13

clean-condition training are 18.02%, 15.71% and 22.86% for THEQ, PHEQ and QHEQ, respectively. In the case of multi-condition training, the average WER results for these three HEQ approaches are slightly better than that of the MFCC-based baseline system (average WERs of 12.30%, 9.5% and 10.23% for THEQ, PHEQ and QHEQ, respectively) which might mainly be due to the fact that with multi-condition training, the mismatch between the training and test conditions can be reduced to a great extent.

On the other hand, Table 7 shows the average WER results obtained by combining PHEQ with different temporal average (TA) operations of different span orders. When the span order is set to 0, it denotes that only PHEQ was applied to the feature vector components. The results in Table 7 demonstrate that combining PHEQ with anyone of the TA operations can further provide an additional relative WER reduction of about 5% to 8%. In a word, the TA operations conducted after HEQ indeed provide a good compensation for non-stationary noises. Nevertheless, TA operations with much higher span orders may instead result in the degradation of the recognition performance.

Table 7. Average WER results (%) obtained by combining PHEQ with different TA operations of different span orders.

		Span Order					
		0	1	2	3	4	5
Clean-Condition Training	Non-Causal MA	15.71	14.57	14.53	15.78	16.61	16.87
	Causal MA	15.71	15.20	14.88	14.66	14.61	15.06
	Non-Causal ARMA	15.71	14.55	14.41	14.94	15.11	15.21
	Causal ARMA	15.71	14.52	14.49	14.86	15.00	16.72
Multi-Condition Training	Non-Causal MA	9.5	8.96	8.98	9.66	10.18	10.75
	Causal MA	9.5	9.35	9.22	8.98	8.95	9.08
	Non-Causal ARMA	9.5	8.92	8.86	9.04	9.13	9.18
	Causal ARMA	9.5	9.22	8.87	8.87	9.25	9.34

4.3 Comparison with Other Normalization Approaches

Finally, we compare the above HEQ approaches with the conventional normalization approaches. The average WER results for the MFCC-based baseline system, as well as for CMS and CMVN, for both clean- and multi-condition training, are shown in Table 8 and presented graphically in Figures 3 and 4, respectively. Notice that the results for THEQ, PHEQ and PHEQ-TA were obtained with the best settings from the above experiments. GHEQ is the recognition results obtained using a Gaussian probability distribution with zero mean and unity variance as the reference distribution rather than using the probability distributions of the entire training data as the reference distributions [Torre *et al*. 2005]. In other words, each feature space dimension is normalized to a standard normal distribution. It can be found that all the HEQ approaches provide significant performance boosts over the MFCC-based baseline system, and they are also better than CMS and CMVN for both clean- and multi-condition training. If TA is further applied after CMVN (*i.e*., MVA) or PHEQ (*i.e*., PHEQ-TA), the recognition results of MVA or PHEQ-TA will be considerably better than those obtained by using CMVN or PHEQ alone.

The experimental results shown in this and the previous sections suggest the following observations:

- The estimation of CDF can have a significant influence on the performance of HEQ. The cumulative-histogram method can give a reliable estimation if there is a large amount of speech feature vectors available; otherwise, the order-statistics based method is recommended.

Table 8. Comparison of the average WER results (%) obtained by the MFCC-based baseline system and various normalization approaches for clean- and multi-condition training.

	Clean-Condition Training				Multi-Condition Training			
	Test A	Test B	Test C	Average	Test A	Test B	Test C	Average
MFCC	47.37	48.42	40.55	45.45	13.56	13.34	17.06	14.65
CMS	26.17	22.06	27.72	25.32	13.27	12.99	13.77	13.34
CMVN	20.21	19.84	21.13	20.39	12.18	11.23	13.21	12.21
MVA	16.63	14.92	17.90	16.48	8.86	8.82	9.69	9.12
THEQ	18.13	16.41	19.51	18.02	11.97	11.47	13.44	12.30
GHEQ	17.69	15.59	18.70	17.32	9.00	8.73	9.60	9.11
PHEQ	15.91	14.43	16.80	15.71	9.23	8.89	10.38	9.50
QHEQ	23.74	21.73	23.11	22.86	8.91	10.03	11.75	10.23
PHEQ-TA	14.29	13.75	15.20	14.41	8.72	8.64	9.21	8.86

- The full cumulative distribution function matching approach, such as THEQ, GHEQ, or PHEQ, gives better recognition performance than the quantile-corrective approach, such as QHEQ.
- In contrast, assuming that the probability distributions of speech feature vectors will follow Gaussian distributions (*e.g.*, GHEQ), the transformation functions used in PHEQ are directly learned from the observed distributions of speech feature vectors. As the results show in Table 8, PHEQ outperforms all the other equalization approaches in most cases for clean-condition training.
- The performance of GHEQ appears slightly better than PHEQ for multi-condition training. This result is probably explained by the fact that multi-condition training can substantially reduce environmental mismatch. Consequently, normalizing the speech feature vectors into a standard normal distribution or normalizing a distribution learned from the training speech seems to make no significant difference in multi-condition training.
- Performing TA after HEQ is necessary, because TA can alleviate the influence of sharp peaks and valleys that were caused by some non-stationary noises or occurred during the equalization process.

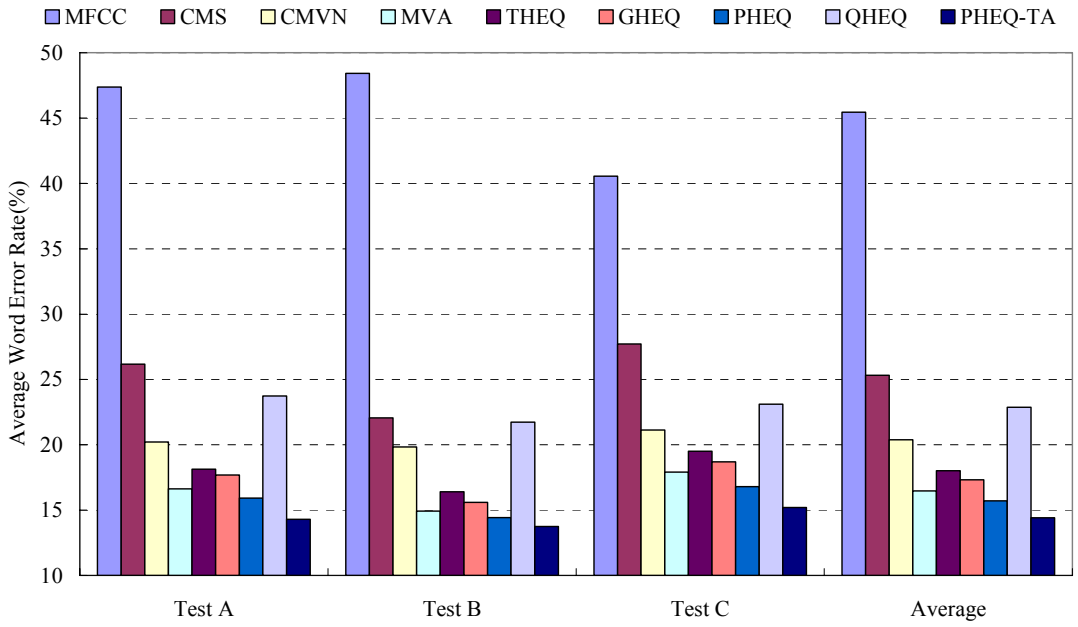


Figure 3. Average WER results (%) obtained by the MFCC-based baseline system and various normalization approaches for clean-condition training.

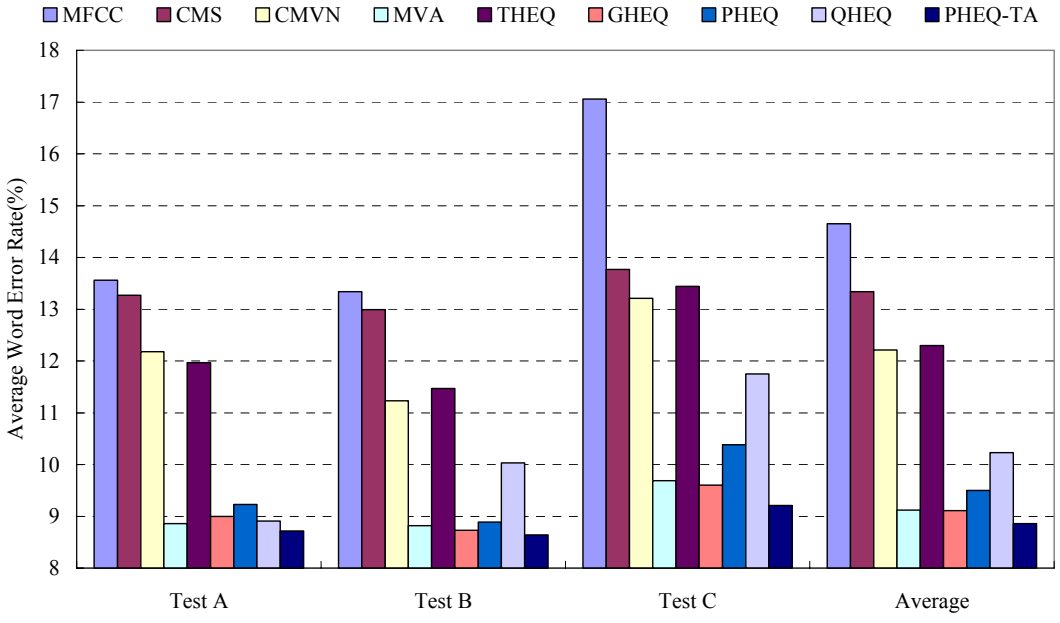


Figure 4. Average WER results (%) obtained by the MFCC-based baseline system and various normalization approaches for multi-condition training.

Table 9. A summary of storage requirement and computational complexity with respect to different HEQ approaches.

Method	Storage Requirement	Computational Complexity
THEQ	Large - depending on the number of reference pairs kept in the look-up table	Medium - depending on the look-up table size for searching the corresponding restored value
QHEQ	Small - depending on the number of quantiles for quantile-correction	High - depending on the value ranges and resolutions of parameters for online grid search.
PHEQ	Small - depending on the order of the polynomial functions	Low - depending on the order of the polynomial function

4.4 Storage Requirement and Computational Complexity

As mentioned in the previous sections, the HEQ approaches have some drawbacks for practical implementation issues, such as requiring large storage consumption and high computational cost, which might make them infeasible when being applied to ASR systems with limited storage and computation resources. Therefore, in this subsection, we analyze these HEQ approaches from two perspectives: the storage requirement and the computational complexity.

In general, the number of reference pairs $\left(C_{Train}(y), \bar{y}_{B_i}\right)$ kept in the look-up table for THEQ cannot be too small. As indicated in Table 1, the recognition performance for the Aurora 2 task will not saturate until the table size is large than 1,000. If 1,000 reference pairs are kept with double precision for THEQ, it requires a memory space of about 1M bytes to store the transformation table for the equalization of all dimensions of the feature vectors. However, for other complicated recognition tasks, such as large vocabulary continuous speech recognition (LVCSR) of broadcast news, it normally requires a much larger size of look-up table to keep the feature transformation/equalization information for better recognition performance, which also implies the need of much larger storage consumption. However, for QHEQ, a small number of quantiles (usually the number is set to 4) is enough for the efficient transformation of speech feature vectors. The storage requirement of QHEQ is very small when compared to THEQ. Similarly, the storage requirement of PHEQ depends mainly on the order of the polynomial functions. In the case of using the polynomial functions with the order set to seven, it roughly requires a memory space of 2.5K bytes to store the coefficients of the polynomial functions.

On the other hand, the computational complexity of THEQ is mainly determined by the size of the look-up table. As the reference pairs $\left(C_{Train}(y), \bar{y}_{B_i}\right)$ stored in the look-up table increase, the complexity for searching the corresponding restored value $\bar{y}_{B_i}$ for the input $C_{Train}(y)$ would become much higher even though the table-lookup procedure can be implemented with the hash table or other efficient data structures. When QHEQ is being used

in the test phase, its computational complexity is the highest when compared to the other two HEQ approaches (THEQ and PHEQ), which is due to the fact that an exhaustive online grid search is required for finding the optimum transformation parameters α and γ. The search process is completely dominated by the value ranges of α and γ, and the resolutions, *i.e.*, the step sizes for updating the values, of α and γ. In contrast to the above two approaches, the computational complexity of PHEQ is almost negligible. It requires only a few mathematical operations, which will result in a tremendous saving in the computational cost. A summary of storage requirement and computational complexity is shown in Table 9.

5. CONCLUSIONS

In this paper, we have given a detailed review of various histogram equalization (HEQ) approaches for improving ASR robustness. Three approaches, namely, the table-based histogram equalization (THEQ), the quantile-based histogram equalization (QHEQ) and the polynomial-fit histogram equalization (PHEQ), were extensively compared and analyzed, in terms of the recognition performance, storage requirement and computational complexity. Moreover, the usage of temporal average (TA) operations also has been investigated for alleviating the influence of sharp peaks and valleys caused by some non-stationary noises or noises occurring during equalization. It has been found that PHEQ outperforms the other equalization approaches and it only requires a small amount of storage consumption and computational cost. The best results were obtained by combing PHEQ with TA that was in the form of non-causal auto-regression moving average. Relative word error rate reductions of 68% and 40% over the MFCC-based baseline system have been obtained for clean- and multi-condition training, respectively.

Acknowledgements

This work was supported in part by the National Science Council, Taiwan, under Grants: NSC 96-2628-E-003-015-MY3 and NSC95-2221-E-003-014-MY3.

REFERENCES

Acharya T. and A. K. Ray, "Image Processing: Principles and Applications," Wiley-Interscience, 2005.

Buera, L., E. Lleida, A. Miguel, A. Ortega and O. Saz, "Cepstral Vector Normalization Based on Stereo Data for Robust Speech Recognition," *IEEE Transaction on Audio, Speech and Language Processing*, 15(3), 2007, pp. 1098-1113.

Chen, C.-P. and J. Bilmes, "MVA Processing of Speech Features," *IEEE Trans. on Audio, Speech and Language Processing*, 15(1), 2007, pp. 257-270.

Dharanipragada, S. and M. Padmanabhan, "A Nonlinear Unsupervised Adaptation Technique for Speech Recognition," *In Proceedings of the 6th International Conference on Spoken Language Processing(ICSLP 2000)*, Beijing, China, 2000.

Ephraim, Y. and D. Malah, "Speech Enhancement Using a Minimum Mean-Square Log-Spectral Amplitude Estimator," *IEEE Transaction on Acoustic, Speech and Signal Processing*, 33(2), 1985, pp. 443-445.

Furui, S., "Cepstral Analysis Techniques for Automatic Speaker Verification," *IEEE Transaction on Acoustic, Speech and Signal Processing*, 29(2), 1981, pp. 254-272.

Gales, M. J. F., "Maximum Likelihood Linear Transformations for HMM-based Speech Recognition," *Computer Speech and Language*, 12(2), 1998, pp. 75-98.

Gauvain, J.-L. and C.-H. Lee, "Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains," *IEEE Transaction on Speech and Audio Processing*, 2(2), 1994, pp. 291-297.

Gong, Y., "Speech Recognition in Noisy Environments: A Survey," *Speech Communication*, 16(3), 1995, pp. 261-291.

Hermansky, H and N. Morgan, "RASTA Processing of Speech, " *IEEE Transaction on Speech and Audio Processing*, 2(4), 1994, pp. 578-589.

Hilger, F. and H. Ney, "Quantile Based Histogram Equalization for Noise Robust Speech Recognition, " *In Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech 2001)*, Aalborg, Denmark, 2001.

Hilger, F. and H. Ney, "Quantile Based Histogram Equalization for Noise Robust Large Vocabulary Speech Recognition," *IEEE Transactions on Audio, Speech and Language Processing*, 14(3), 2006, pp. 845-854.

Hirsch, H. G. and D. Pearce, "The AURORA Experimental Framework for the Performance Evaluations of Speech Recognition Systems under Noisy Conditions," *In Proceedings of the 6th International Conference on Spoken Language Processing(ICSLP 2002)*, Beijing, China, 2002.

Hsu, C.-W. and L.-S. Lee, "Higher Order Cepstral Moment Normalization (HOCMN) for Robust Speech Recognition," *In Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP 2004)*, Quebec, Canada, 2004.

Hsu, C.-W. and L.-S. Lee, "Extension and Further Analysis of Higher Order Cepstral Moment Normalization (HOCMN) for Robust Features in Speech Recognition," *In Proceedings of the 9th International Conference on Spoken Language Processing (ICSLP 2006)*, Pittsburgh, Pennsylvania, 2006.

Huang X., A. Acero, H. Hon, "Spoken Language Processing: A Guide to Theory, Algorithm and System Development," Prentice Hall, 2001

Huo, Q., C. Chany and C.-H. Lee, "Bayesian Adaptive Learning of the Parameters of Hidden Markov Model for Speech Recognition," *IEEE Transaction on Speech and Audio Processing*, 3(4), 1995, pp. 334-345.

Junqua, J. C., J. P. Haton and H. Wakita, "Robustness in Automatic Speech Recognition," Kluwer, 1996.

Lee, L.-S. and B. Chen, "Spoken Document Understanding and Organization," *IEEE Signal Processing Magazine*, 22(5), 2005, pp. 42-60.

Leggeter, C. J. and P. C. Woodland, "Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Models," *Computer Speech and Language*, 9, 1995, pp. 171-185.

Lial M., R. N. Greenwell and N. P. Ritchey, "Calculus with Applications," Addison Wesley, 2005.

Lin, S.-H., Y.-M. Yeh and B. Chen, "Exploiting Polynomial-Fit Histogram Equalization and Temporal Average for Robust Speech Recognition," *In Proceedings of the 9th International Conference on Spoken Language Processing (ICSLP 2006)*, Pittsburgh, Pennsylvania, 2006.

Molau, S., D. Keysers and H. Ney, "Matching Training and Test Data Distributions for Robust Speech Recognition," *Speech Communication*, 41(4), 2003, pp. 579-601.

Molau, S., "Normalization in the Acoustic Feature Space for Improved Speech Recognition," Ph.D. Dissertation, Computer Science Department, RWTH Aachen University, Aachen, Germany, 2003.

Molau, S., F. Hilger and H. Ney, "Feature Space Normalization in Adverse Acoustic Conditions," *In Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, Hong Kong, 2003.

Moreno, P., "Speech Recognition in Noisy Environment," Ph.D. Dissertation, ECE Department, Carnegie Mellon University, Pittsburgh, PA, 1996.

Segura, J. C., C. Benitez, A. Torre, A. J. Rubio and J. Ramirez, "Cepstral Domain Segmental Nonlinear Feature Transformations for Robust Speech Recognition," *IEEE Signal Processing Letters*, 11(5), 2004, pp. 517-520.

Suk, Y. H., S. H. Choi and H. S. Lee, "Cepstrum Third-Order Normalisation Method for Noisy Speech Recognition," *Electronics Letters*, 35(7), 1999, pp. 527-528.

Torre, A., A. M. Peinado, J. C. Segura, J. L. Perez-Cordoba, M. C. Bentez and A. J. Rubio, "Histogram Equalization of Speech Representation for Robust Speech Recognition," *IEEE Transactions on Speech and Audio Processing*, 13(3), 2005, pp. 355-366.

Vikki, A. and K. Laurila, "Segmental Feature Vector Normalization for Noise Robust Speech Recognition," *Speech Communication*, 25, 1998, pp. 133-147.

Wu, J. and Q. Huo, "An Environment-Compensated Minimum Classification Error Training Approach Based on Stochastic Vector Mapping," *IEEE Transactions on Audio, Speech and Language Processing*, 14(6), 2006, pp. 2147-2155.

Young, S., G. Evermann, M. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, "The HTK Book (for HTK Verson 3.3)," Cambridge University Engineering Department, Cambridge, UK, 2005.