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:

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

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:

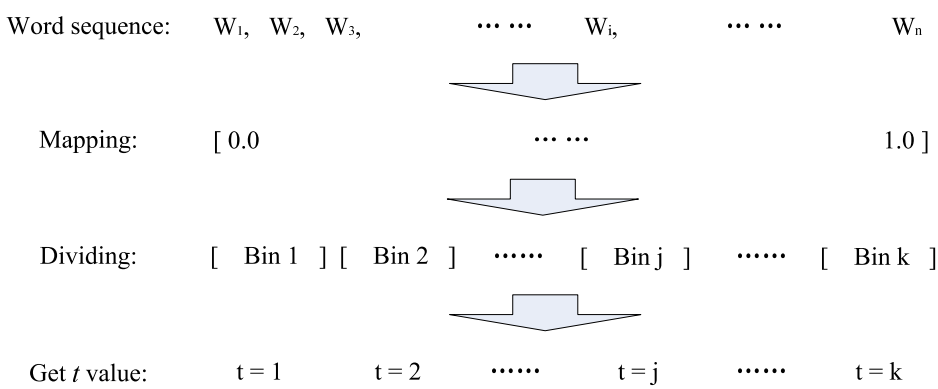


Figure 1. Calculation of the t value in NS ngram model

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.

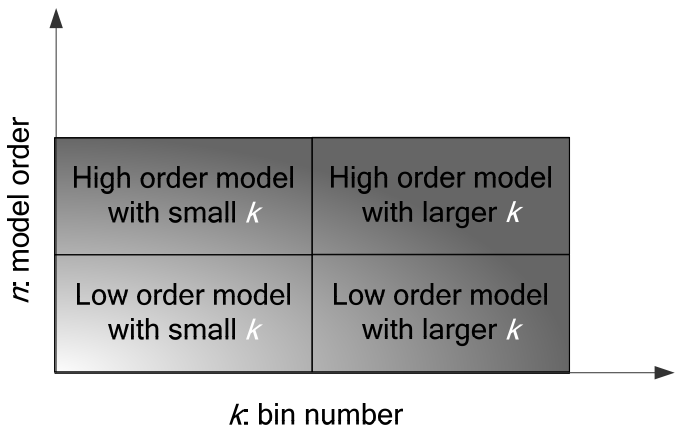


Figure 2. Data sparseness problem in NS ngram model

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.

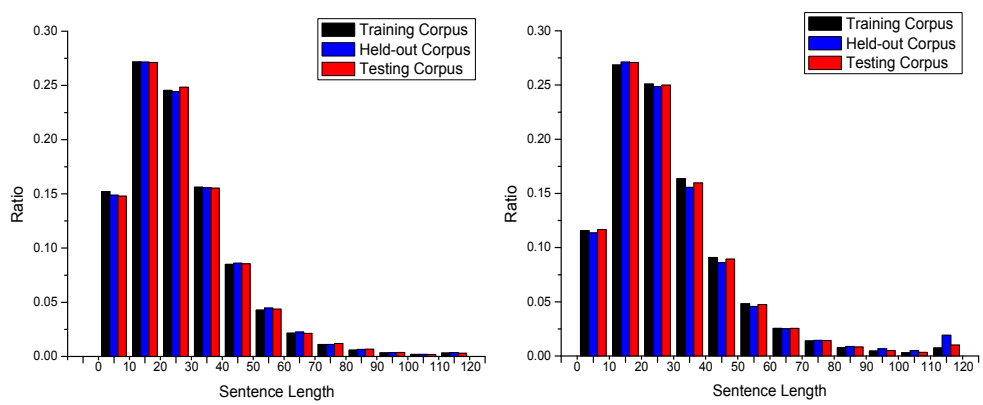
Text Corpus

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 6^{th} month 6.29×10^5	2/3 of 6^{th} month 1.25×10^6
People's Daily corpus in 2000	1-11 months 2.27×10^7	1/3 of 12^{th} month 7.01×10^5	2/3 of 12^{th} 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.

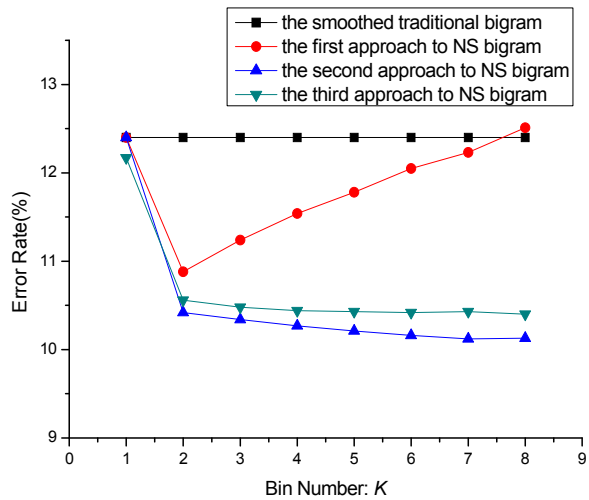


Figure 4. Comparison of the three smoothing approaches

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.