7659, England, UK
Journal of Information and Computing Science
Vol. 2, No. 3, 2007, *pp*. 215-222

Exploiting Word Positional Information in Ngram Model for Chinese Text Input Method *

Jinghui Xiao+, Bingquan Liu and Xiaolong Wang

School of Computer Science and Techniques, Harbin Institute of Technology, Harbin, 150001, China

(*Received 24 November 2006, accepted 30 January 2007*)

Abstract. This paper aims to improve the performance of the Pinyin-to-Character Conversion system which is the core of Chinese text input method. The ngram model is the current solution to the Pinyin-to-Character Conversion system. This paper enhances the traditional ngram model by relaxing its stationary hypothesis and exploiting the word positional information. The Non-stationary ngram (NS ngram) model is proposed. Several related issues are discussed in detail, including the formal definition, the model implement, the training algorithm and the space complexity of the NS ngram model. Evaluated on the Pinyin-to-Character Conversion task, the NS ngram model outperforms the traditional ngram model significantly with great error rate reductions. Meanwhile, the training algorithm presented in this paper can estimate the parameters in the NS ngram model effectively and efficiently.

Keywords: ngram model, stationary hypothesis, Pinyin-to-Character Conversion, Chinese text input method

1. Introduction

This paper aims to improve the performances of the Pinyin-to-Character Conversion system which is the core of Chinese text input method.

The standard keyboard is initially designed for the native English speaker. In Asia, such as China, Japan, Thailand and so on, people can not input their language into computer directly by the standard keyboard. When user is to input the language, he first input the sequence of the phonetic symbols on the standard keyboard. Then these symbols are converted into the desired character sequence by the software called 'IME' which is the abbreviation of 'Input Method Editor' [1]. Pinyin is the standard phonetic symbol of Chinese and the above process is usually called the Pinyin-to-Character Conversion task in China. The task is the core of the Chinese IME software.

The Pinyin-to-Character Conversion task can also be taken as a simplified automatically speech recognition task [2]. 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, the techniques proposed in this paper are also illuminating to the speech recognition task.

The ngram model is the current solution to the Pinyin-to-Character Conversion task. The Markov hypothesis is made on the ngram model so as to simplify the probability inference. There are actually two hypotheses implied by the Markov hypothesis [3], named the limited history hypothesis and the stationary hypothesis. The first one assumes that the probability of the current word is only determined by a few of previous words, but irrelevant to the whole history of word. The second one assumes that the above probability is irrelevant to the actual position of word in the sentence. The most obvious extension to the traditional ngram model is simply to relax the limited history hypothesis and move to higher-order ngram model [4]. But the high-order ngram model suffers from the dimension curse problem [5] which hampers its further applications. Bigram model and trigram model are currently two prevalent ngram models.

From another point of view, this paper relaxes the stationary hypothesis and enhances the traditional

* This investigation is supplied by the project of National Natural Science Foundation "Research on the non-stationary property of language element in natural language processing" (No. 60673037) and by the key project of National Natural Science Foundation "Research on theory and technique of Question-Answering information retrieval" (No. 60435020).
+ Corresponding Author: Tel, 86-0451-86413322, *Email Address*: xiaojinghui@insun.hit.edu.cn

ngram model by exploitation of the word positional information. It is based on the philosophy that the current word is not only determined by its contextual words, but also relevant to its position in the sentence. For example, the phrase of “first of all” is usually used to start a sentence, but rarely occurs elsewhere in a sentence. Then higher probability should be assigned to this phrase by the ngram model when it occurs in the front of a sentence, and lower probability elsewhere. 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 the sentence. Therefore, the ngram model can benefit from modelling the word positional information.

In this paper, the Non-stationary ngram (NS ngram) model is proposed by relaxing the stationary hypothesis of the traditional ngram model. The word positional information is exploited so as to improve the performance. Several related issues are discussed in detail, including the formal definition, the model implement, the training algorithm and the space complexity of the NS ngram model. Experiments are carried out on the Pinyin-to-Character Conversion task. Compared with the traditional ngram model, great error rate reductions have been achieved by the NS ngram model.

This paper is structured as follows. The traditional ngram model is briefly reviewed in section 2. The NS ngram model, together with its related issues, is presented in section 3. Experimental results are provided in section 4. The related works are described in section 5 and the conclusions are drawn in section 6.

2. Brief Review of Ngram Model

The ngram model is one of the most popular language models. A language model is to determine the probability of a sequence of words. The probability is usually decomposed as the product of the conditional probability of word which composes of the sequence. For a word sequence $S = w_{l_1,p_1} w_{l_2,p_2} ... w_{l_n,p_n}$, its probability is determined by

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

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

The ngram model makes the Markov hypothesis to simplify the formula (1) so as to make it tractable. The procedures are described as below:

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

Actually, there are two hypotheses implied by the Markov hypothesis [3]:

1. Limited history hypothesis: the probability of the current word is dependent only on the previous *n-1* words, but irrelevant to the whole history of word.
2. Stationary hypothesis: the ngram probability is determined only by the words which compose of the probability, but irrelevant to their positions where they possess in the sequence *S*.

The formula (1) is firstly simplified by the limited history hypothesis, resulting in the second item in the formula (2). Then the stationary hypothesis is applied and the final form of the ngram model is obtained, as indicated by the last item of the formula (2). Since the conditional probability is irrelevant to the word position, we use w_{l_i} to denote w_{l_i,p_j} .

3. Non-stationary Ngram Model

In this section, the NS ngram model is firstly defined formally. Then its implement are discussed in detail, including the representation of the word positional information and the concrete form of its probability function. Thirdly, the training algorithm is presented. Lastly, the space complexity is analyzed.

3.1. Definition of NS Ngram Model

As analyzed in the section 1, the occurrences of words are relevant to their positions where they possess in a sentence. It’s beneficial to exploit the positional information to determine the probability of word. Markov hypothesis is too restricted to exploit the positional information due to its stationary assumption. This paper relaxes the stationary hypothesis of the traditional ngram model and proposes the NS ngram model. The formula is presented 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, the formula (1) is simplified only by the limited history hypothesis, rather than the stationary hypothesis. Then the word conditional probability is not only determined by the history words, but also by their positions in the sentence *S*. A single variable *t* is used to denote the positional information in the conditional probability. The ngram model is a special case of the NS ngram model in which *t* is a constant.

To implement the NS ngram model, three fundamental questions should be answered:

- How to represent the positional variable of *t* formally?
- How to define the concrete form of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$?
- How to estimate the parameters in $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$?

In the following sections, the above three questions are discussed in detail and the solutions are provided.

3.2. Representation of *t*

Since *t* denotes the word positional information, a natural way to represent *t* is to take the word position index in the sentence as the concrete value of *t*. But there are two severe problems in this method. Firstly, index has different meanings in the sentence of different length. For example, there are two sentences: "Yesterday I saw you." and "Yesterday I saw you were walking around here". In both sentences, the word "you" is assigned to the same position index --- 4. However, the word "you" appears at the *end* of the first sentence, while it appears in the *middle* of the second. The word "you" has completely different positional information in the two sentences. We can't tell the difference by the above definition of *t*. Secondly, since a sentence may have arbitrary length, the word position index can be any natural number. However, computer can not deal with infinite value.

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

This paper divides the above range of [0, 1] into several equivalent classes (bins) and each class share the same positional information. The value of *t* is set to the class index.

More formally, the value of *t* in the NS ngram model can be calculated by the following procedures:

1. Calculate the ratio of the word position index and the length of sentence for each word. The word positional information is then mapped into the range of [0, 1].
2. Divide the range of [0, 1] into *k* bins. Words in each bin share the same positional information.
3. Set the *t* value of each word as the index of the bin where the word falls in.

The figure 1 shows an example of the above procedures:

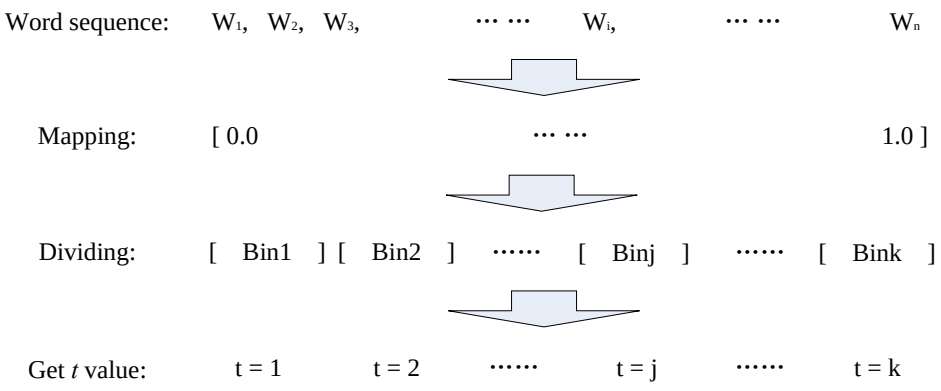

Fig 1: Calculation the *t* value in NS ngram model

The more numbers of bins, the more positional information are extracted from the sentence. The normal ngram model is a special case of the NS ngram model in which there is only one bin.

3.3. Form of Probability Function

Based on the representation of *t*, this section discusses the concrete form of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$. Obviously, $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$ is the function of *t*. In this paper, $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t)$ is further defined as the function of the statistical variables of *t*. Two kinds of statistical variables have been constructed: the expectation of *t* and the variance of *t*. Such an assumption is made that more probability should be assigned to the current word if its positional information fits better with the positions in the training corpus, and less probability vice versa. Concretely, the probability function form is defined as below:

$$p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}, t) = \frac{1}{Z} e^{\frac{\alpha \times V(w_{l_i})}{((t - E(w_{l_i}))^2 + \beta)}} \times p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}) \tag{4}$$

The notations in the formula (4) are described as below:

- The positional information of the current word w_{li}: t
- The expectation of position of w_{li} in training corpus: $E(w_{li})$
- The variance of position of w_{li} in training corpus: $V(w_{li})$
- The probability of the traditional ngram model: $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}})$
- The coefficients of α and β
- The normalizing factor of Z which is defined by the following formula:

$$Z = \sum_{t=1}^{t=k} e^{\frac{\alpha \times V(w_{l_i})}{((t - E(w_{l_i}))^2 + \beta)}} \times p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}) \tag{5}$$

where *k* is the number of bins so as to calculate the value of *t*.

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 the formula (4). As the value of $t\text{-}E(w_{li})$ decreases, *t* fits for the training corpus better and more probability should be awarded. Henceforth, the probability function is descendent with the value of $t\text{-}E(w_{li})$ as the formula (4) shows. Moreover, the probability 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 the sentence. Then it's unreasonable to decrease the weight just as the term $t\text{-}E(w_{li})$ increases. In such a situation, the value of item $V(w_{li})$ of the *active* word is usually bigger than that of the *inactive* words. Then it can provide a balance for the value of $t\text{-}E(w_{li})$.

3.4. Form of Probability Function

This section discusses how to estimate the value of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}})$. As the formula (4) presents, the probability function of the NS ngram model can be decomposed into two parts: the traditional ngram probability and the adjusted weight for it. Accordingly, the training process of the NS ngram model is also divided into two procedures, separately to estimate the optimal value of each part. Concretely speaking, the traditional ngram probability of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}})$ is firstly estimated. Then the adjusted weight is optimized so as to get the best performance.

The probability of $p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}})$ can be estimated under the Maximum Likelihood Estimation (MLE) principle, as described by the following formula:

$$p(w_{l_i} \mid w_{l_{i-n+1}} ... w_{l_{i-1}}) = \frac{C(w_{l_{i-n+1}} ... w_{l_i})}{C(w_{l_{i-n+1}} ... w_{l_{i-1}})} \tag{6}$$

where $C(w_{l_{i-n+1}} ... w_{l_i})$ is the co-occurrence frequency of the word sequence $w_{l_{i-n+1}} ... w_{l_i}$. When *n* is large, the data sparseness problem is prone to occur. Some proper smoothing techniques [6-8] can be utilized to solve that problem.

In the part of the adjusted weight, there are totally four factors to be determined: α and β for the universal function; $E(w_{li})$ and $V(w_{li})$ for each word w_{li}. Among them, $E(w_{li})$ and $V(w_{li})$ can be estimated from the training corpus. The formulas are presented as below:

$$E(w_{li}) = \frac{1}{N(w_{li})} \sum_{t} t(w_{li}) \tag{7}$$

where $t(w_{li})$ is the concrete positions of the word w_{li} in the training corpus; $N(w_{li})$ is the total frequency of the word w_{li} in the training corpus.

$$V(w_{li}) = \frac{1}{N(w_{li})} \sum_{t} (t(w_{li}) - E(w_{li}))^2 \tag{8}$$

However, it's hard to get the optimal values of α and β in a direct way. This paper optimizes the values of α and β by the genetic algorithm on the held-out corpus. The genetic algorithm 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

3.5. Space Complexity

This section analyzes the space complexity of the NS ngram model and compares it with the traditional ngram model. Some notations are defined for convenience:

- The total number of word in the lexicon: l
- The model order: n

In the traditional ngram model, all the parameters are devoted to store the ngram probability. The space complexity of the n-order ngram model is $O(l^n)$. In the NS ngram model, besides the ngram probability, the statistical variables of the positional information also require the storage. Totally two kinds of statistical variable are exploited for each word. The space complexity of the n-order NS ngram model is $O(l^n + 2l)$. In realities, n is usually beyond *2*. In that case, $2l$ is far less than l^n and it can be neglected in the term of $O(l^n + 2l)$. Then the space complexity of the NS ngram model is $O(l^n)$ which is exactly as same as the traditional ngram model.

4. Experiments and Discussions

This section presents the experimental results of the NS ngram model on the Pinyin-to-Character Conversion task. The data set is firstly described. Then the performances of the NS ngram model are presented. Finally, the effectiveness of the genetic algorithm is investigated in the training process of the NS ngram model.

4.1. Data Set Description

This paper chooses the half year of the People's Daily corpus in 1998 as the text corpus in the experiments. The text corpus is divided into three parts: the training corpus which consists of the first five months' corpus, the held-out corpus which is one of the third corpus of the 6th month, and the test corpus which consists of the rest corpus. The detailed information is presented in table 1 as below:

Tab 1. Description of Text Corpus

	Training Corpus	Held-out Corpus	Test Corpus
Number of months	*1-5 months*	*1/3 of 6th month*	*2/3 of 6th month*
Number of characters	9.09×10^6	0.63×10^6	1.25×10^6

The pinyin corpus is also necessary for the NS ngram model to be evaluated on the Pinyin-to-Character Conversion task. When the model is evaluated, the pinyin corpus is firstly converted into the text corpus. Then the converted results are compared with the standard text corpus and the error rate is calculated. We get the pinyin corpus from the text corpus by a conversion toolkit whose precision is beyond 98%. Since the pinyin corpus is not a golden corpus, the error in the pinyin corpus could lead to the conversion error of the NS ngram model on the Pinyin-to-Character Conversion task. Therefore, the actual error rate is lower than the reported results in this paper. The performance of the real system is a little better than the experimental results. However, because of the high precision of our conversion toolkit, there are not many errors in the pinyin corpus. Thereby, the reported results can be regarded to be close enough to the actual performance of the NS ngram model.

4.2. Performances of NS ngram Model

This section evaluates the NS ngram model on the Pinyin-to-Character Conversion task. All the experiments are carried out on the test data. The NS models with variant number of bins are investigated. The traditional ngram model is taken as the baseline model. The experimental results are presented in table 2 and figure 2.

Tab 2. Performances of the NS ngram Model

Bin Number		**K=1**	**K=2**	**K=3**	**K=4**	**K=5**	**K=6**	**K=7**	**K=8**
Ngram	**Error Rate**	12.17%	---	---	---	---	---	---	---
NS ngram	**Error Rate**	12.17%	10.56%	10.48%	10.44%	10.43%	10.42%	10.43%	**10.4%**
	Reduction	0%	13.23%	13.89%	14.22%	14.30%	14.38%	14.30%	**14.54%**

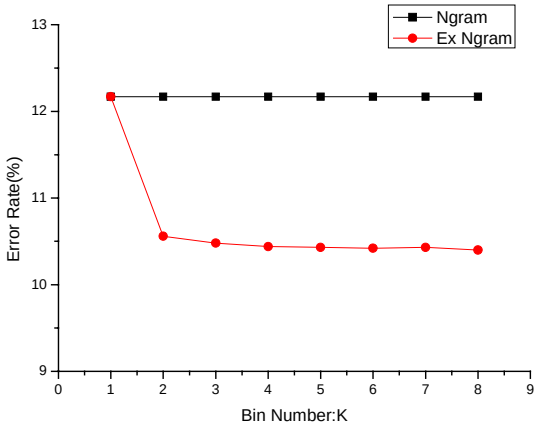

Fig 2 Performances of the NS ngram Model

In the above experiments, the NS ngram model outperforms the traditional ngram model significantly. As much as 14.54% error rate reduction has been achieved. That fact proves that the NS ngram model has much more predictive capability than the traditional ngram model. Moreover, as k increases, the error rate of the NS ngram model decreases constantly. That indicates that the improvements of the NS ngram model are due to the increasing positional information. Lastly, the NS ngram model performs stably after $k=2$, which indicates that a small number of bins are enough to estimate the statistical variables in the NS ngram model.

4.3. Effectiveness of Genetic Algorithm

This section investigates the effectiveness of the genetic algorithm in the training process of the NS ngram model. The settings of the genetic algorithm are presented as below:

Tab 3. Settings of 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 pinyin-to-character converter

The figure 3 presents the optimizing process of the NS ngram model on the held-out corpus by the genetic algorithm.

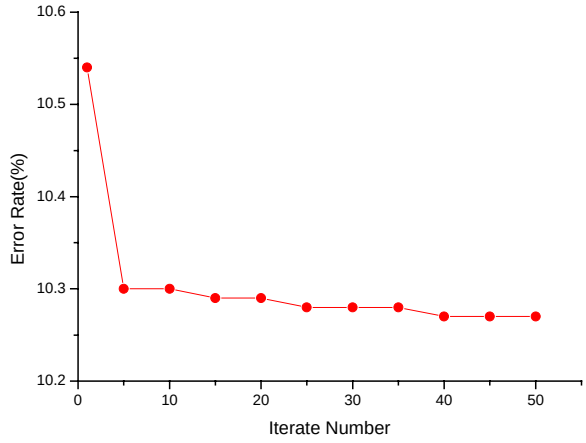

Fig 3 Error Rates of the NS ngram Model on Held-out Corpus

In the above figure, the error rate of the NS ngram model decreases along with the iterative number of the genetic algorithm. That fact proves the genetic algorithm can optimize parameters of the NS ngram model effectively. Moreover, the genetic algorithm converges quickly. After about twenty iterations, the error rate of the Pinyin-to-Character Conversion system becomes stable, which indicates that the genetic algorithm is efficient to get the optimal values of parameters in the NS ngram model. In a word, the genetic algorithm performs effectively and efficiently in the training process of the NS ngram model.

5. Related Works

There are many ways to improve the performance of the ngram model. The most obvious one is to relax the limited history hypothesis and to build up the high-order ngram model, which has been discussed in the section 1. Another simple method is to construct the skipping ngram model [9, 10], in which the current word is conditioned on the skipped words in the history, rather than the adjacent words. Together with the traditional ngram model, the skipping model can exploit more history words and avoid the dimension curse problem at the same time. In experiments, the skipping model can yield limited improvements by interpolating with the traditional ngram model.

Class-based ngram model [11] is constructed based on word cluster instead of single word. Some syntax and semantic information can be captured in this way, and in the meanwhile, the parameter space is reduced greatly and the data sparseness problem is alleviated. However, the predictive ability of Class-based ngram model is much lower than that of the traditional ngram model. It usually achieves improvements by interpolating with the traditional model.

Caching ngram model [12, 13] assumes that people tends to use as few words as possible in an article.

Therefore, if a word has been used, it will possibly be used again in the future. Caching ngram model usually is used to construct a self-adaptive ngram model.

6. Conclusions

This paper enhances the traditional ngram model so as to improve the performance of the Pinyin-to-Character Conversion system which is the core of Chinese text input method. The stationary hypothesis is relaxed and the positional information of word is exploited. Based on that, the Non-stationary ngram model is proposed and several related issues are discussed in detail, including the formal definition, the model implement, the training algorithm and the space complexity of the NS ngram model. From the experimental results, several conclusions can be drawn:

1. Compared with the traditional ngram model, the NS ngram model has achieved great error rate reduction on the Pinyin-to-Character Conversion task by exploitation of the positional information of word.
2. The genetic algorithm performs effectively and efficiently in the training process of the NS ngram model.

7. References

[1] http://www.microsoft.com/globaldev/handson/user/IME_Paper.mspx

[2] Jianfeng Gao, Hao Yu and Wei Yuan, et al. Minimum Sample Risk Methods for Language Modeling. *Human Language Technology Conference on Empirical Methods in Natural Language Processing* (HLT/EMNLP 2005), Vancouver, Canada. Oct 6-8, 2005

[3] Christopher D. Manning and Hinrich Schutze. *Foundation of Statistic Natural Language Processing*. The MIT Press. 1999.

[4] Bob Carpenter. Scaling high-order character language models to gigabytes. *Proceedings of the 2005 Assocation for Computational Linguistics Software Workshop*.

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

[6] Irving J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*. 1953, **40**(16): 237-264.

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

[8] F. Jelinek and R. L. Mercer. Interpolated estimation of Markov source parameters from sparse data. *Pattern Recognition in Practice*. 1980, 381-397.

[9] R. Rosenfeld. *Adaptive statistical language modeling: a maximum entropy approach*. Ph.D. dissertation. Carnegie Mellon Univ., Pittsburgh, PA., 1994.

[10] Hermann Ney, Ute Essen, and Reinhard Kneser. On structuring probabilistic dependences in stochastic language modeling. *Computer, Speech, and Language*. 1994, **8**:1-38.

[11] Brown, F. Peter, Vincent J. Della Pietra and Peter V. deSouza et. al. Class-based n-gram models of natural language. *Computational Linguistics*. 1992, **18**(4): 467-479.

[12] Roland Kuhn. Speech Recognition and the Frequency of Recently Used Words: A Modified Markov Model for Natural Language. *12th International Conference on Computational Linguistics* (COLING88). Budapest, August 1988, 348-350.

[13] Roland Kuhn and Renato De Mori. A Cache-Based Natural Language Model for Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 1990, **12**(6): 570–583..