

# A Segment-based Hidden Markov Model for Real-Setting Pinyin-to-Chinese Conversion

Xiaohua Zhou<sup>1</sup>, Xiaohua Hu<sup>1</sup>, Xiaodan Zhang<sup>1</sup>, Xiajiong Shen<sup>2</sup>

College of Information Science & Technology, Drexel University, Philadelphia, PA, USA<sup>1</sup>

School of Computer and Information Engineering, Henan University, China<sup>2</sup>

xiaohua.zhou@drexel.edu, {thu, xzhang}@ischool.drexel.edu

## Abstract

Hidden markov model (HMM) is frequently used for Pinyin-to-Chinese conversion. But it only captures the dependency with the preceding character. Higher order markov models can bring higher accuracy, but are computationally unaffordable to average PC settings. We propose a segment-based hidden markov model (SHMM), which has the same magnitude of complexity as first-order HMM, but generates higher decoding accuracy. SHMM tells a word from a bigram connecting two words, and assigns a reasonable probability to words as a whole. It is more powerful than HMM to decode words containing over two characters. We conduct a comprehensive Pinyin-to-Chinese conversion evaluation on Lancaster corpus. The experiment shows the perfect sentence accuracy is improved from 34.7% (HMM) to 43.3% (SHMM). The one-error sentence accuracy is increased from 72.7% to 78.3%. Furthermore, SHMM can seamlessly integrate with pinyin typing correction, acronym pinyin input, user-defined words, and self-adaptive learning all of which are a must for a commercial Pinyin-to-Chinese conversion product in order to improve the efficiency of pinyin input.

## Categories and Subject Descriptors

I.2.7 [Artificial Intelligence]: Natural Language Processing—language models

## General Terms

Algorithm, Experimentation, Performance

## Keywords

Chinese Input, Pinyin, Segment-based Hidden Markov Model

## 1. Introduction

Chinese character is graph-based rather than letter-based so that it can not be input to computer directly. Transliteration input and structure-based input are two major approaches to Chinese input. The latter approach such as “Wu Bi” needs a long learning time and is used by a small number of PC users. The most basic implementation of transliteration input is based on Pinyin-to-

Chinese conversion. There are 406 pinyin syllables corresponding to over 6,000 common Chinese characters. Therefore, it is low efficient to convert Pinyin to Chinese character one by one because one syllable can map to 15 characters on average. A more efficient solution is to input more Pinyin syllables one time and make the best use of character dependency to decode. Thus, the task of Pinyin-to-Chinese conversion is similar to speech recognition and can be solved by language modeling.

First-order hidden markov model (HMM) is frequently used for Pinyin-to-Chinese conversion. It achieves a good balance between conversion accuracy and conversion speed. For example, the pinyin “zhong guo” can generate about 66 candidate bigrams, but HMM can easily get the correct conversion “中国” (China) because this bigram is much often than others in real texts. However, HMM only captures the dependency with the preceding character and does not work very well on converting trigrams and four-grams. Unfortunately, there are a large number of three- or four-character words, idioms and proper names in Chinese. This problem can be overcome by using a trigram language model or higher order HMM without complexity and memory constraints. Unfortunately, most PCs are constrained by memory and unaffordable to such models which generate a large state space and need a large state transition matrix.

For this reason, we propose an alternative solution referred to as segment-based hidden markov model (SHMM) to solve the problem in this paper. The idea of SHMM is that we distinguish between Chinese words and bigrams connecting two words when computing the likelihood of a Chinese text. We compile a Chinese lexicon and search a segment through it. If the segment does exist, we treat it as the product of bigrams (i.e. the state change is markovian), otherwise assign it as a whole a probability according to another separate scheme (i.e. the state change within the segment is not markovian). On one hand, using two separate schemes, SHMM can easily handle three- or four-character words. On the other hand, a lexicon of about 200K words, proper names, and idioms ranging from two to four characters can cover almost all frequently used and popular words in various domains, while storing and searching such a small lexicon brings only a little extra workload to an average PC.

We conduct a Pinyin-to-Chinese conversion evaluation on Lancaster corpus [4]. The experiment shows that the perfect sentence accuracy is improved from 34.7% (HMM) to 43.3% (SHMM). The one-error sentence accuracy is increased from 72.7% to 78.3%. The character accuracy is improved from 82.9% to 86.1%. In addition, SHMM can seamlessly integrate with pinyin typing correction, acronym pinyin input, user-defined words, and self-adaptive learning all of which are a must for a commercial

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'07, November 6--8, 2007, Lisboa, Portugal.

Copyright 2007 ACM 978-1-59593-803-9/07/0011...\$5.00.

Pinyin-to-Chinese conversion product in order to improve the efficiency of pinyin input.

## 2. Background of Chinese Pinyin Input

Pinyin-to-Chinese conversion is getting very important as the number of Chinese users is increasing at an exponential rate with the development of internet. Many commercial products, such as “Microsoft Pinyin”, “Google Pinyin”, “Sogou Pinyin”, “Chinese Star”, “Ziguang Pinyin”, and “Pinyin Jiajia”, are available to Chinese PC users. Among these products, the first four support sentence level pinyin input and the last two only support phrase-level pinyin input. The state-of-the-art performance is about 10%~20% character level error rates [2].

Statistical language model is widely used for Pinyin-to-Chinese conversion [1] [2]. In [1] [2], a unified trigram-based approach was proposed and the reported conversion accuracy reached 95%, much higher than the accuracy of commercial products. In order to make the system usable in real word, they also proposed a typing model which could correct common typing errors. However, they did not report their evaluation settings and protocols clearly in the paper.

Statistical language models including bigram and trigram, however, can not capture long-range dependency as well as multiple interacting features. One solution to this problem is post-pruning. Zhang et al. proposed a rule-based post-pruning approach in [7]. They manually built a knowledge base which contained both syntactic and semantic rules. In their experiment, such a pruning technique can significantly improve the conversion rate. However, the testing sample was quite small. The reliability and effectiveness of this approach need further validation.

Zhao et al. [8] proposed another maximum entropy based approach to handle long-range dependencies. ME can incorporate both local features such as preceding characters and long-range dependencies. Trigger pairs is used as long-range dependency features. The best ME model could reduce about 4% error compared to a trigram model in their experiment. In the paper, they presented the model of how to predict a single character according to the context, but did not show the details of how to efficiently search the whole optimal sequence. Thus, the complexity of this approach is unclear though the approach itself is very attractive.

## 3. Segment-based Hidden Markov Model

Given a sequence of pinyin syllables  $S$ , the task of Pinyin-to-Chinese conversion is to find a sequence of Chinese characters  $C$ , which maximizes the conditional probability  $p(C|S)$ . Using Bayesian law, we have:

$$\begin{aligned} C &= \arg \max_C p(C|S) \\ &= \arg \max_C \frac{p(S|C)p(C)}{p(S)} \\ &= \arg \max_C p(S|C)p(C) \end{aligned} \quad (3.1)$$

The first term  $p(S|C)$  is the typing (or pinyin) model and the second term  $p(C)$  is the language model [1].

The typing model is relatively simple. Without consideration of typing error and polyphone, it can be rewritten as:

$$p(S|C) = \prod_i p(S_i|C_i) \quad (3.2)$$

where  $S_i$  and  $C_i$  are the  $i$ -th pinyin and character respectively. The language model  $p(C)$  is the prior probability of  $C$ . Conceptually, one can try all possible  $C$  and select the one maximizing the joint probability  $p(S,C)$ . Practically, some strong assumptions such as first-order markov property are made in order to simplify the problem. With first-order markov assumption, one can use Viterbi algorithm to efficiently search the optimal  $C$ .

Now we formally introduce the notation, HMM and SHMM. In the task of Pinyin-to-Chinese conversion, a Chinese character corresponds to a state in HMM and pinyin is the observation. Suppose,  $N$  is the total number of Chinese characters in the dictionary and  $h_i$  stands for the  $i$ -th character (state).  $S$  is the input pinyin and contains  $T$  syllables.  $S_t$  denotes the pinyin observation at time  $t$ . Let  $\delta_t(i)$  denote the best score (highest probability) for the first  $t$  pinyin observations ending with the character (state)  $h_i$ . Then we have the recursive equation for HMM [5]:

$$\delta_{t+1}(i) = \max_{1 \leq j \leq N} [\delta_t(j)p(h_i|h_j)]p(S_{t+1}|h_i) \quad (3.3)$$

In first order HMM, the state (character) at time  $t$  is always transitioned from the state (character) at time  $t-1$ . We relax this limitation in SHMM and allow non-markovian state transition within a time segment. After and before the segment, the state transition is still markovian. In other words, the first state after the segment depends on the last state of the segment and the first state of the segment depends on the last state of the preceding segment. Formally, we define  $w_{m,i}$  as a segment with time length  $m$  (containing  $m$  characters) and ending with state (character)  $h_i$ , and define  $S_{m,t}$  as the observations (pinyin) from time  $t-m+1$  to  $t$ , and define  $M$  as the maximum length of a segment. Then we have the recursive equation:

$$\delta_{t+1}(i) = \max_{1 \leq m \leq M, 1 \leq j \leq N} [\delta_{t+1-m}(j)p(w_{m,i}|h_j) \times p(S_{m,t+1}|w_{m,i})] \quad (3.4)$$

The equation (3.4) looks a little complicated at this moment. Let us ignore the typing model and use an example to illustrate motivation behind SHMM. Suppose we need to compute the prior probability of a text containing four characters  $ABCD$ . In SHMM, a sequence of characters is always segmented into words before computing its prior probability. Given the text  $ABCD$ , there are different possible segmentations. For instance,  $AB|CD$  denotes this text is the connection of two two-character words. Basically, the outer loop controlled by the variable  $m$  (the length of the segment) in equation (3.4) tries to enumerate all possible segmentations.

The implementation of the segment-based typing model is relatively simple. We compile a Chinese lexicon containing words, proper names, and idioms, and sort all entries by their pinyin. Given a pinyin segment observation  $S_{m,t+1}$ , we search it through the lexicon. If the segment  $w_{m,i}$  exists, then  $p(S_{m,t+1}|w_{m,i})$  is set to one, else it has zero probability.

The computation of the state transition probability within a segment is application specific. In order to avoid long n-grams which always generate a huge number of parameters, we provide a light-weight implementation for Pinyin-to-Chinese conversion. In general, given the word  $w$  beginning with character  $h_s$  and the last

character of it preceding word  $h_e$ , we have the following approximation:

$$p_s(w|h_e) \approx p(h_s|h_e) \frac{N(w)}{N(h_s)} \quad (3.5)$$

$N(w)$  and  $N(h_e)$  denote the frequency of the word  $w$  and the character  $h_e$  in the training corpus, respectively. From the equations (3.4) and (3.5), we can conclude that SHMM only need to maintain an extra list of words and their pinyin and frequency as well as some additional computation cost which is linear to the complexity of HMM. The additional computation cost is determined by the maximum segment length. For Chinese language, the maximum segment length can be set to four because there are very few words containing five or more characters. Therefore, the additional computation cost is really not much. In our testing (see Section 4.4), we find out the conversion speed for SHMM is almost same as for HMM because the bottleneck lies in the retrieving transition probabilities from the sparse matrix, which is same to both SHMM and HMM.

In order to improve the efficiency of pinyin input and offer better user experience, commercial pinyin products often provide some extra features including pinyin typing correction, self-adaptive learning, acronym pinyin input, and user-defined words. Like other statistical language models, SHMM can easily integrate pinyin typing correction by adding pinyin correction rules to the typing model, and support self-adaptive learning by interpolating the background bigram model with the user bigram model. SHMM is based on words rather than characters and thus can seamlessly integrate acronym pinyin input by adding pairs of pinyin acronym and word to the lexicon. Acronym pinyin input is an efficient way to input Chinese word by inputting the initial of pinyin instead of full pinyin. For instance, most commercial products can get “中国” (China) by only typing “z g”, which is the initial of the full pinyin “zhong guo”. However, acronym input makes pinyin more ambiguous and comes out more candidates. Most commercial products including those based on HMM can not handle acronym pinyin in a unified model and thus can not utilize the context to filter out candidates. SHMM deals with acronym pinyin and full pinyin in a unified approach which makes SHMM more attractive than HMM to real Chinese input applications.

In summary, SHMM attempts to segment a sequence of pinyin into words and then model words and bigrams connecting two adjacent words in two different schemes. Such a separation makes SHMM overcome some problems of the traditional bigram language models such as underestimation of three-character and four character words and overestimation of some fake n-grams. In addition, SHMM can integrate acronym pinyin input in a unified approach. All these benefits are acquired at the cost of a little extra computational cost and memory.

## 4. Experiment Settings and Result Analysis

### 4.1 Lexicon and Training Corpus

SHMM needs a bigram language model and a list of words and their frequency learned from the training corpus. We collected diverse Chinese web pages from Internet and built a corpus containing about 500 million Chinese characters. We trained the bigram language model on this corpus. We compile the Chinese lexicon from several sources. The main source is Xinhua Dictionary, which is an authoritative dictionary about Chinese

characters and words and collects about 120K words containing 2-4 characters. We also developed an algorithm to automatically collect from the training corpus new words which are not listed in Xinhua Dictionary. The number of extra words collected is about 80K. We employed a maximum match algorithm [6] to count the frequency of Chinese words in the training corpus.

### 4.2 Testing Corpus

We evaluate the Pinyin-to-Chinese conversion on the Lancaster corpus [4]. Lancaster is a Chinese corpus with pinyin and POS annotated and word segmented. The documents in the corpus are collected from Internet and have a good diversity. We split texts into sentences using punctuations as delimiters. After excluding sentences containing less than four characters, or containing numbers or non-Chinese characters, there are 7,732 remaining sentences for testing. The shortest sentences and longest sentences contain four characters and thirty-six characters, respectively. All testing sentences have a total number of 78,622 characters and an average sentence contains about ten characters. For some polyphonic Chinese characters, the Lancaster corpus did not annotate their pinyin correctly. We correct them as much as possible.

### 4.3 Experimental Results

There is little formal study of Pinyin-to-Chinese decoding evaluation in literature, not alone sentence-level evaluation. For this reason, we propose three accuracy metrics in this paper. They are *character accuracy*, *perfect sentence accuracy*, and *one-error sentence accuracy*. The first metric, character accuracy, is very straightforward. It is the percentage of corrected Chinese characters in total decoded characters. The perfect sentence accuracy is defined as the percentage of corrected sentences in total sentences to decode. A sentence is corrected if all characters in a sentence are correctly decoded. We emphasize sentence level accuracy because it is time consuming for users to correct errors within a long sentence. One-error sentence accuracy is defined as the percentage of sentences containing up to one word-level error (zero or one error) in total sentences to decode.

Both HMM and SHMM need a bigram language model. In the experiment, we use Jelinek-Mercer [3] approach to smoothing the bigram language models. JM smoothing linearly interpolate the bigram model with the unigram language model. We empirically tune the mixture weight and find out that when the weight for bigram model is set to 0.9, both HMM and SHMM have best results. The comparisons of HMM and SHMM are shown in Table 1. SHMM significantly outperform HMM in terms of all three accuracy metrics. Especially, the perfect sentence accuracy and one-error sentence accuracy obtain considerable improvement, which will offer better experience to users because for long sentence conversion, it is time-consuming for users to correct even one mis-decoded characters in the sentence.

**Table 1. The comparison of segment-based hidden markov model (SHMM) to standard hidden markov model (HMM) on the task of Pinyin-to-Chinese decoding**

Metric	Character Accuracy	Perfect Sent. Accuracy	One-error Sent. Accuracy
HMM	82.9%	34.7%	72.7%
SHMM	86.1%	43.3%	78.3%
Improvement	3.9%	24.8%	7.7%

We examine the conversion results and list in Figure 1 some representative sentences which SHMM successfully converted but HMM failed. In the first example, the first four characters form an idiom in Chinese. In the second example, the first three characters are the name of a famous communist party member in China. The first order HMM can only trace back to the preceded character and thus fails in both examples. SHMM searches the lexicon and finds out they are words rather independent characters and then assigns them a reasonable probability as a whole and eventually decode the pinyin. Compared to first order HMM, SHMM is quite good at decoding out three-character or four-character words.

The third example demonstrates the problem of overestimated bigram. The bigram “不动” (do not move) is also a part of two frequently occurred Chinese words “不动产” (real property) and “不动声色” (hiding one’s emotion) . When the bigram is a part of some words or idioms, they do not function as a bridge connecting two words. But the usual bigram language model does not differentiate these cases. As long as two characters are next to each other, they will be counted as a bigram. Thus, the bigram is overestimated in the sense that a bigram should be used to connect two words. In SHMM, the counts of a bigram as a part of a word or idiom are removed. In the third example, we can see that the correct answer bubbles up after fixing the overestimated bigram problem.

<b>Pinyin:</b> mei shi mei ke guan xin zhe wo
<b>HMM:</b> 美食 (gourmet) 每刻(every minute) 关心(take care) 着我 (me)
<b>SHMM:</b> 每时每刻 (every minute) 关心 (take care) 着我 (me)
<b>Pinyin:</b> jiao yu lu lie shi ling yuan
<b>HMM:</b> 教育(education) 路 (road) 烈士陵园 (cemetery)
<b>SHMM:</b> 焦裕禄 (a person’s name) 烈士陵园 (cemetery)
<b>Pinyin:</b> bu dong ji suan ji
<b>HMM:</b> 不 (not) 动 (move) 计算机 (computer)
<b>SHMM:</b> 不懂 (don’t know much) 计算机 (computer)

**Figure 1. Examples which SHMM correctly converted but HMM failed.**

#### 4.4 Running Performance

About 3.5 million bigrams are learned from training corpus and stored in a sparse matrix. In the experiment, the bigram matrix is not fully loaded into memory, but caches up to 2,000 rows (corresponding to about 12M size of peak memory use). The lexicon contains about 200K Chinese words, proper names, and idioms each of which ranges from two characters to four characters. The lexicon is loaded into memory and consumes about 6M size of memory. Plus other consumptions, the peak memory consumption for HMM and SHMM are about 14M and 20M, respectively. SHMM take another 6M size of memory compared to HMM.

The Pinyin-to-Chinese conversion tool is implemented in Java and the experiment is conducted on a Windows XP PC. SHMM takes about 436 seconds to convert all 7,732 sentences in Lancaster corpus while HMM takes about 383 seconds. Surprisingly, the actual running time for SHMM is quite close to HMM. Looking at equation (3.3) and (3.4), we can see that SHMM pays some extra computation cost for enumerating all possible segmentations. However, the bottleneck of decoding is not in this procedure, but in retrieving the bigram count from the

sparse matrix, and therefore that extra computation cost does not make big difference on actual conversion time.

## 5. Conclusions and Future Work

In this paper, we proposed a novel segment-based hidden markov model (SHMM) and successfully applied it to the application of Pinyin-to-Chinese conversion. First-order HMM is frequently used to model Chinese language. However, it is not very effective in modeling Chinese words, proper names, and idioms containing more than two characters. SHMM automatically segments a sequence of pinyin or characters into words, and then models words and bigrams connecting two adjacent words in two different schemes, and thus overcome partially the weakness of HMM. In addition, SHMM is able to integrate some features such as acronym pinyin input in a unified way, making itself more attractive than HMM to real-setting Pinyin-to-Chinese conversion applications. All of these benefits are achieved at a little extra memory and computation cost.

## 6. Acknowledgement

This work is supported in part by NSF Career Grant IIS 0448023, NSF CCF 0514679, PA Dept of Health Tobacco Settlement Formula Grant (No. 240205 and No. 240196), and PA Dept of Health Grant (No. 239667). We also thank Xuguang Zhang who helped us type testing sentences into three commercial products during evaluation.

## 7. References

- [1] Z. Chen, K. F. Lee, A new statistical approach to Chinese pinyin input. In *ACL-2000*, Hong Kong, 2000, 241-247.
- [2] Jianfeng Gao, Hai-Feng Wang, Mingjing Li, Kai-Fu Lee. 2000. A Unified Approach to Statistical Language Modeling for Chinese. *IEEE, ICASSP 2000*.
- [3] Jelinek, F. And Mercer, R. Interpolated estimation of markov sourceparameters from sparse data, *Pattern Recognition in Practice*, E. S. Gelsema and L. N. Kanal, Eds., 1980, pp. 381-402.
- [4] A. McEnery, Z. Xiao, The Lancaster Corpus of Mandarin Chinese: A Corpus for Monolingual and Contrastive Language Study, *LREC 2004 Proceedings*, pp. 1175-1178
- [5] Rabiner, L.R., A tutorial on hidden Markov models and selected applications, In *IEEE proceedings of speech recognition*, 77(2), Feb, 1989:257 – 286
- [6] P. K. Wong, C. K. Chan, Chinese word segmentation based on maximum matching and word binding force , in *Proc. 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 1996, pp. 200-203
- [7] Y. Zhang, B. Xu, C. Zong, Rule-based Post-Processing of Pinyin to Chinese Characters Conversion System. *ISCSLP*, 2006
- [8] Y. Zhao, X. Wang, B. Liu and Y. Guan, Research of Pinyin-To-Character conversion based on Maximum Entropy model, *Journal of Electronics(China)*, 23(6), November, 2006, pp 864-869