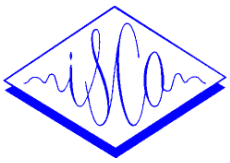

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

INPUT CHINESE SENTENCES USING DIGITS

Fang Zheng, Jian Wu and Wenhu Wu

Center of Speech Technology, State Key Laboratory of Intelligent Technology and Systems,
Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
fzheng@sp.cs.tsinghua.edu.cn, http://sp.cs.tsinghua.edu.cn/

ABSTRACT

Chinese character input is always a key issue in a variety of Chinese based applications especially when only a small number keypad is available. Though many kinds of Chinese character encoding schemes are proposed according to Chinese character characteristics, such as the shape, they are not straightforward and will take users a long time to learn. An easy way is to input via Chinese pinyins. In this paper, we establish the mapping between digit string and pinyin as well as the mapping between the pinyin string and the word, referred to as the Syllable-Digit search Tree (SDT) and the Word-Syllable search Tree (WST) respectively. By using these two search trees as well as the word N-gram language model and the syllable-synchronous network search (SSNS) algorithm, any digit string can be easily converted into Chinese word sequence or sentence. Without users' selecting from candidates, the character error rate (CER) of digit-to-character (D/C) conversion is 6.6% across a test text consisting 22,083 characters.

1. INTRODUCTION

Finding solutions to fast Chinese character input is a key goal for Chinese language processing researchers. Large-vocabulary continuous Chinese speech recognizers, Chinese spoken dialogue systems and optical character recognizers are often regarded as very useful methods for Chinese character input. In such systems, there are almost two layers for processing; one is known as the acoustic speech recognition or character recognition while another is the word decoding. The decoding stage is often different for different languages and it is often the focus for human-machine interface research and development.

TABLE 1. A TYPICAL NUMBER KEYPAD LAYOUT

1	**2** *abc*	**3** *def*
4 *ghi*	**5** *jkl*	**6** *mno*
7 *pqrs*	**8** *tuv*	**9** *wxyz*
#	**0** **<space>**	*****

In case we have only a number keypad such as in mobiles and telephones, Chinese character input will be difficult. Researchers have proposed many kinds of Chinese character encoding schemes based on the shapes of characters, where users have to spend a lot of time learning them. Inputting characters via pinyins is a good solution because it will not take much time thinking about how to type in pinyins for most people who are quite familiar with the standard Chinese pinyin scheme. In this case, we only need to provide a letter-to-digit mapping scheme.

Table 1 is a typical number keypad by which we can input letters via digit, where 3~4 letters share, or are encoded into, one digit. This is the letter level Code-Duplication Problem (CDP). Thus to input a letter, you have to press the corresponding digit key once, twice, thrice or four times according to the position that the letter appears on the corresponding digit key. For example, four presses are needed to input letter 'z'.

In summary, the following steps are needed to input a Chinese character: (1) input each letter of a Chinese pinyin by pressing corresponding digits to form a Chinese syllable and (2) select the desired Chinese character that shares this pinyin. For example, to input character '中', first input pinyin '*zhong*' by typing in '9999 44 666 66 4' and then select it from the candidate list '中重种众终钟仲肿忠衷...' and to input '国', input '*guo*' by typing in '4 88 666' and the select it from '国过果郭裹涡锅帼馘聒...'. This is really a tedious process.

A good question is whether it is possible to input a letter by pressing the shared digit key only once where keystrokes will be reduced and then convert the pinyin string into character string. For example, encoding the pinyins '*zhong*' and '*guo*' into '94664' and '486' respectively, then we will have '中国'. There are two kinds of CDPs, the pinyin level CDP, for instance, '*zhong*' and '*xiong*' share '94664', and the character level CDP, for example, many characters share '*zhong*'.

To solve these CDPs, we present a solution that adopts a word N-gram Chinese language model. Experimental results across a big testing database show this conversion system achieves a satisfying conversion performance.

In this paper, we study the language modeling in details. In Section 2, we present our proposed Digit-based Chinese Input Method Engine (IME), including the syllable-digit search tree and the word-syllable search tree structure, the N-gram language modeling, the search strategies and so on. All these modules are finally integrated to form a D/C conversion system and in Section 3 the experimental results are given. Conclusions will be drawn in Section 4.

2. DIGIT-BASED INPUT METHOD ENGINE

Our Digit-Based IME tries to find the most likely word sequence W^* given the input digit string D among all possible word sequences W's that match the digit string D. By Bayes' rule,

$$W^* = \arg\max_W P(W \mid D) = \arg\max_W P(D \mid W)P(W) \qquad (1)$$

where $P(D)$ is discarded because it is constant for the given digit string and it does not affect the final conversion result.

$P(W)$ is given by the word N-gram language model. $P(D|W)$ is the conditional probability of the digit string D given the word sequence W, hence it can be expressed as

$$P(D \mid W) = \begin{cases} 1, & D = C(W) \\ 0, & D \neq C(W) \end{cases} \quad (2)$$

where $C(W)$ is the concatenated digit string of the word sequence W. Substituting Equ. (2) into Equ. (1), we obtain,

$$W^* = \arg\max_{W:D=C(W)} P(W) \quad (3)$$

This implies that the D/C conversion should be performed as the word decoding procedure under the constraint $D=C(W)$.

2.1 LEXICAL TREES

The D/C conversion decoding with $D=C(W)$ can be described by a digit lexicon organized in a tree structure called word-digit tree where each word is represented by a digit string. Nevertheless this big tree is not practical to use because it takes a long time to rebuild it whenever the user modifies the vocabulary.

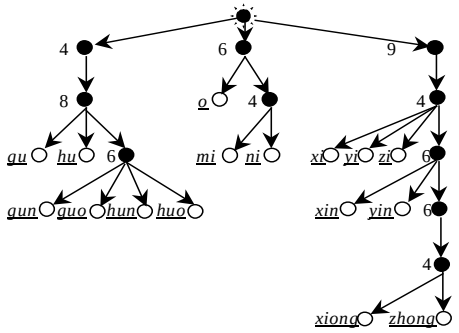

FIGURE 1. PART OF THE SYLLABLE-DIGIT TREE

A two-level lexical tree (TLLT) structure is proposed to solve this problem. The lower-level tree is called a Syllable-Digit Tree (SDT) as illustrated in Fig. 1. This tree is always fixed. Any digit string traveling from root to leaf constantly will result in a pinyin string.

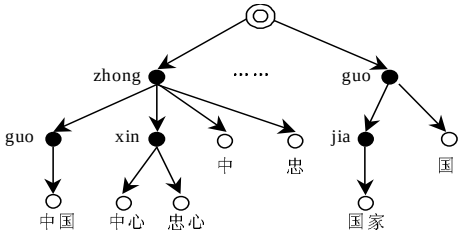

FIGURE 2. PART OF THE WORD SYLLABLE TREE

The higher-level tree is referred to as the Word-Syllable Tree (WST)[6]. This lexical tree will be much smaller than word-digit tree and hence it almost takes no time to be incrementally rebuilt when the user modify the vocabulary.

Using these two trees, the decoding constrained $D=C(W)$ can be interpreted as $\{C_S(W)=S\} \cap \{C_S(S)=D\}$ where $C_S(W)$ is the concatenated syllable string of the word sequence W and $C_D(S)$ is the concatenated digit string of the syllable sequence S.

2.2 ENHANCED BACK-OFF LANGUAGE MODEL

Any given definite Chinese syllable string $S = \{s_1, \cdots, s_L\}$ could correspond to many Chinese word string candidates due to the uncertainty of the Chinese word boundaries[6]. Among these word string candidates $\{W \mid C_S(W) = S\}$, we should take the one with the highest likelihood score as the final conversion result for the given syllable string. Assume a word string is $W = \{w_1, \cdots, w_N\} \overset{def}{=} W_1^N$, its score is defined as the probability of the word string, which is simplified as

$$P(W) \approx P(w_1)P(w_2|w_1)\prod_{n=3}^{N} P(w_n|w_{n-2}, w_{n-1}) \quad (4)$$

This is the well-known tri-gram model. In order to give every unseen word pair a reasonable probability, a new proposal is made in this paper. We will take the bi-gram model as an example to give the idea of our proposed method[5].

2.2.1 Good-Turing and Katz smoothing methods

In the Good-Turing (GT) re-estimation, the frequency (also known as the occurring *count*) of any seen n-gram is discounted and the accumulated residual probability is re-distributed to all the unseen n-grams[3], shown as follows,

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \quad (4)$$

where r is the frequency of certain n-gram, r^* is the frequency after discounted and n_r is the number of n-grams occurring exactly r times in the training data. After normalization, the relative frequency

$$P_{GT}\left(W^{(r)}\right) = \frac{r^*}{\sum_{r=0}^{\infty} n_r r^*} \quad (5)$$

is taken as the re-estimated probability for word pair $W^{(r)}$ occurring r times. Accordingly, all the zero-frequency bi-grams can be approximated from the relative frequency of those bi-grams with single occurrence, each of which is called a singleton. Katz developed a modified version of this estimation by combining higher-order models and lower-order models when re-estimating the probability of unseen bi-grams, which is actually a kind of back-off model[4]. The Katz Smoothing (KS) supposes that the occurring probability of the zero frequency bi-gram is also associated with the focused word, or the current observed word, in the word pair. The KS's idea can be expressed as follows,

$$P_{KS}\left(w_i|w_{i-1}\right) = \begin{cases} d_r\left(w_{i-1}\right) f\left(w_i|w_{i-1}\right), & r > 0 \\ \alpha\left(w_{i-1}\right) P_{KS}\left(w_i\right), & r = 0 \end{cases} \quad (6)$$

where

$$f\left(w_i|w_{i-1}\right) = \frac{c\left(w_{i-1}, w_i\right)}{c\left(w_{i-1}\right)} = \frac{r}{c\left(w_{i-1}\right)} \quad (7)$$

$$d_r = \begin{cases} 1, & r > T \\ \dfrac{r^*/r - (k+1)n_{k+1}/n_1}{1-(k+1)n_{k+1}/n_1}, & 0 < r \le T \end{cases} \quad (8)$$

and

$$\alpha(w_{i-1}) = \frac{1 - \sum_{w:c(w_{i-1},w)>0} P_{KS}(w|w_{i-1})}{1 - \sum_{w:c(w_{i-1},w)>0} P_{KS}(w)}. \quad (9)$$

In these equations, $c(e)$ is the occurring count of the event e and T is a predefined threshold for this count.

However, the disadvantage of the KS method is that n-gram discount is based on different order n-gram information, depending on whether the n-gram count is zero or non-zero, which may lead to results contradictory to our expectation in a few cases.

2.2.2 Integration of deleted interpolation into KS

The KS can also be expressed in terms of the interpolated model [1][2]

$$P_{KS}(w_i|w_{i-1}) = [\lambda(w_{i-1},w_i)]P_{GT}(w_i|w_{i-1}) + [1-\lambda(w_{i-1},w_i)]P_{KS}(w_i) \quad (10)$$

Obviously, the key difference between the back-off model and the interpolated model is that the later considers the information from lower-order n-gram distributions whereas the former does not when re-estimating the probabilities of n-grams with non-zero counts. In fact, the weight λ in the KS method must be chosen such that

$$\lambda(w_{i-1},w_i) = \begin{cases} 1, & r>0 \\ \dfrac{\alpha(w_{i-1})-1}{\beta(w_{i-1},w_i)-1}, & r=0 \end{cases} \quad (11)$$

where

$$\beta(w_{i-1},w_i) = \frac{1 - \sum_{w:c(w_{i-1},w)>0} P_{KS}(w|w_{i-1})}{n_0 \cdot P_{KS}(w_i)} \quad (12)$$

Based on the interpolated form of back-off model, lower-order n-gram distributions can be easily integrated into the KS re-estimation by changing the formula of the weight during re-estimation of the probabilities of seen n-grams. The modified equation can be rewritten in a uniform formula as

$$\lambda(w_{i-1},w_i) = \begin{cases} \mu(w_{i-1},w_i), & r>0 \\ \dfrac{\alpha(w_{i-1})-1}{\beta(w_{i-1},w_i)-1}, & r=0 \end{cases} \quad (13)$$

In order for the probabilities of all unseen bi-grams to be unaltered after the Katz Smoothing, the weight function $\mu(w_{i-1},w_i)$ must be such that the accumulated probability of the seen word pair is unchanged. This constraint can be expressed as

$$\sum_{w:c(w_{i-1},w)=r} \{[\mu(w_{i-1},w)]P_{GT}(w|w_{i-1}) + [1-\mu(w_{i-1},w)]P_{KS}(w)\} = \sum_{w:c(w_{i-1},w)=r} P_{GT}(w|w_{i-1}), \quad \forall r>0 \quad (14)$$

In general, we can define the weight function in Equation (13) as a linear interpolated formula of $c(w_i)$ and $r = c(w_{i-1},w_i)$ as

$$\mu(w_{i-1},w_i) = \eta_{1r}(w_{i-1}) \cdot c(w_i) + \eta_{2r}(w_{i-1}) \quad (15)$$

For a given word w_{i-1} and the frequency r, by substituting Equ. (15) into Equ. (14), we obtain

$$\eta_{1r}(w_{i-1}) \cdot \sum_{w:c(w_{i-1},w)=r} c(w)[P_{GT}(w|w_{i-1}) - P_{KS}(w)] = (1-\eta_{2r}(w_{i-1})) \cdot \sum_{w:c(w_{i-1},w)=r} [P_{GT}(w|w_{i-1}) - P_{KS}(w)] \quad (16)$$

Therefore given a particular word w_{i-1} and a count r, finding $\eta_{1r}(w_{i-1})$ and $\eta_{2r}(w_{i-1})$ is to maximize

$$\sum_{w:c(w_{i-1},w)=r} c_h(w_{i-1},w) \cdot \{[\mu(w_{i-1},w)]P_{GT}(w|w_{i-1}) + [1-\mu(w_{i-1},w)]P_{KS}(w)\} \quad (17)$$

under the constraints in Equ.s (15) and (16) through Lagrange interpolation formula, where $c_h(w_{i-1}, w)$ is the occurring time of bi-gram (w_{i-1}, w) in the *held-out* data of the training database[2].

The *held-out* algorithm is often used in the Deleted-Interpolation algorithm therefore the modified model is referred to as the Enhanced Katz Smoothing With Deleted Interpolation (EKSWDI).

2.2.3 Simplified EKSWDI

The Enhanced Katz Smoothing With Deleted Interpolation method can be implemented easily in the word decoding procedure. However, it will take a lot of time to calculate all the coefficients, including η_1 and η_2 and a lot of space to store them.

Let us consider the assumptions. First, the Katz Smoothing assumes that the probability of frequent bi-grams estimated under the MLE criterion is adequately reliable, leading to the weights being assigned as constant one with no reduction in performance. Second, as we mentioned above, the lower-order n-gram information should be considered in case r is smaller. According to the two hypotheses, the weight function $\mu(W_{i-1},W_i)$ can be defined as a normalized relative ratio of the bi-gram probability to the uni-gram probability as in Equation (18).

$$\mu_1(w_{i-1},w_i) = \begin{cases} 1, & r>T \\ \dfrac{P_{KS}(w_i|w_{i-1})}{P_{KS}(w_i|w_{i-1}) + P_{KS}(w_i)}, & 0<r\le T \end{cases} \quad (18)$$

This implementation can be explained as the relative confidence measure of the bi-gram probability compared with the uni-gram probability. By this formula, the storage can be saved because the weights can be obtained dynamically in run-time. However, this simplification does not always satisfy the constraint implied in Equation (14). We should re-estimate $\alpha(w_{i-1})$'s and normalize the distributions to make the probability sum be one.

Equation (18) is still a little bit hard to implement, a simpler linear function of the count r is

$$\mu_2(w_{i-1},w_i) = \begin{cases} 1, & r>T \\ r/T, & 0<r\le T \end{cases} \quad (19)$$

This can be used to reduce the consumption time. According to Equ. (19), the recursive expression of modified Katz Smoothing can be generalized as follows

$$P_{MKS}(w_i|w_{i-1}) = \begin{cases} f(w_i|w_{i-1}), & r > T \\ \frac{r}{T}\left(d_r(w_{i-1}) f(w_i|w_{i-1}) - P_{KS}(w_i)\right) + P_{KS}(w_i), & 0 < r \le T \\ \alpha(w_{i-1}) P_{KS}(w_i), & r = 0 \end{cases} \quad (20)$$

The EKSWDI is a back-off model integrated with the interpolation of the low-order n-gram information. It assumes that not only the probabilities of unseen n-grams but also those of seen n-grams could be re-estimated according to the low-order n-gram probabilities, of which the Katz smoothing method can be regarded as one special case.

2.2.4 Choosing the count threshold *T*

For a vocabulary of about 50 words, it would take a very long time to compute the tri-grams. The count threshold *T* should be well chosen. If the number of times a tri-gram W_1^3 occurred in the training text $c(W_1^3)$ is not greater than a constant $T = 5$, we will re-estimate its occurrence probability as a "discount" n consideration of the sparseness of the tri-gram probability matrix. According to our previous experiment, this *Big-Discount* re-estimation not only makes the word decoding procedure much faster, but also improves the decoding accuracy[7].

2.3 SEARCH ALGORITHM

The decoding algorithm used here is the dual-stack syllable synchronous network search (SSNS) algorithm[7], using a compound tree structure of SDT and WST as well as the EKSWDI language model.

The SSNS algorithm together with syllable-digit tree, word-syllable tree and N-gram language model can easily solve the CDP and word segmentation problem in D/C conversion.

By using the WST structure, the fuzzy pronunciation confusion problem (one of the accent problems) can be solved through the arc-splitting technique in the SSNS algorithm.

TABLE 2. EXAMPLES OF FUZZY PRONUNCIATION MAPPING

Whole Syllable		Initial		Final	
ZHI →	JI	Z ↔	ZH	AN ↔	ANG
CHI →	QI	C ↔	CH	EN ↔	ENG
SHI →	XI	S ↔	SH	IN ↔	ING
WANG ↔	HUANG	F ↔	H		
WEN ↔	WENG	N ↔	L		
GUO →	GUI	W ↔	HU		
		Y ←	R		

3. EXPERIMENT

By integrating the above modules, a D/C conversion system named *EasyConv* is established. The vocabulary is designed to include 50,624 Chinese words that are commonly used in Chinese, consisting of 6,201 monosyllable words (12.3%), 37,976 bi-syllable words (75.0%), 1,615 tri-syllable words (3.2%) and 4,832 quad-syllable words (9.5%)[7].

The language model is built on a corpus containing about 200 million Chinese characters. The corpus covers the 4-year's text data of "*People's Daily*" (from 1993 to 1994 and from 1996 to 1997) and a few sections from other Chinese newspaper. The training data are all written texts in news style and in formal language.

The testing data is taken from recent Chinese papers including 1,560 sentences that do not appear in the training corpus, where there are totally 22,083 Chinese syllables (characters). All the testing sentences are labeled with their unique and canonical syllable strings.

We evaluate our method as follows. First, translate the sentences into digital strings according to Table 1 and the labeled syllable strings. Second, take these digit strings as the input of *EasyConv* sentence by sentence. Third, compare the outputted character strings with the corresponding labeled character strings to see how many characters are correctly converted. The character error rate (CER) is 6.6%. The conversion errors are often due to the letter level CDP.

4. CONCLUSION

In this paper, a two-level lexical tree structure is proposed together with the word N-gram language model and the syllable synchronous network search algorithm for digit-to-character conversion. The experimental result shows that this kind of conversion method achieves a satisfying conversion accuracy and hence can be used in applications where only a small number keypad is available for the Chinese character input. For example, it can be used in mobile phones or telephones to input characters. The use of language model results in the following advantages: (1) fewer keystrokes than any other input methods; (2) no need to select in many conversion candidates; and (3) better conversion accuracy. If additional sentence candidates are provided to the user, the accuracy can be even higher in practice. We also intend to reduce the memory requirement to store the language model.

5. REFERENCES

[1] Jelinek F. and Mercer R. L., "Interpolated estimation of Markov source parameters from sparse data", *Pattern Recognition in Practice*, E. S. Gelsema and L. N. Kanal, Eds., 1986, Amsterdam: North-Holland

[2] Jelinek F., "Statistical methods for speech Recognition", The MIT Press, Cambridge, Massachusetts, 1997

[3] Nadas A. "On Turing's formula for word probabilities," *IEEE Trans. on ASSP*, Vol. ASSP-33, No. 6, June 1985

[4] Katz S., "Estimation of probabilities from sparse data for the language model component of a speech recognizer", *IEEE Trans. on ASSP*, 35(3): 400-401, March 1987

[5] Wu J., Zheng F., "On enhancing Katz-smoothing based back-off language model," to appear in proceedings of *International Conference on Spoken Language Processing*, Oct. 2000, Beijing

[6] Zheng F., "A syllable-synchronous network search algorithm for word decoding in Chinese speech recognition", *IEEE International Conference on Acoustic, Speech and Signal Processing* (*ICASSP*), pp. II-601~604, March 15~19, 1999, Phoenix: USA

[7] Zheng F., Wu J., and Song Z.-J., "Improving the syllable-synchronous network search algorithm for word decoding in continuous Chinese speech recognition," to appear in *Journal of Computer Science and Technology*, Sept. 2000