Research Journal of Applied Sciences, Engineering and Technology 7(5): 975-980, 2014
DOI:10.19026/rjaset.7.344
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: January 31, 2013 Accepted: February 25, 2013 Published: February 05, 2014

Research Article

Linear Reranking Model for Chinese Pinyin-to-Character Conversion

[1]Xinxin Li, [1]Xuan Wang, [1]Lin Yao and [1, 2]Muhammad Waqas Anwar
[1]Computer Application Research Center, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China
[2]Department of Computer Science, COMSATS Institute of Information Technology, Abbottabad, Pakistan

Abstract: Pinyin-to-character conversion is an important task for Chinese natural language processing tasks. Previous work mainly focused on n-gram language models and machine learning approaches, or with additional hand-crafted or automatic rule-based post-processing. There are two problems unable to solve for word n-gram language model: out-of-vocabulary word recognition and long-distance grammatical constraints. In this study, we proposed a linear reranking model trying to solve these problems. Our model uses minimum error learning method to combine different sub models, which includes word and character n-gram LMs, part-of-speech tagging model and dependency model. Impact of different sub models on the conversion are fully experimented and analyzed. Results on the Lancaster Corpus of Mandarin Chinese show that our new model outperforms word n-gram language model.

Keywords: Dependency model, minimum error learning method, part-of-speech tagging, word n-gram model

INTRODUCTION

Research on pinyin-to-character conversion is beneficial for many Chinese language processing tasks, such as speech recognition, intelligent inputting methods. However, this task is still difficult due to the existence of homophone. For Chinese characters, there are about 418 different syllables without tones (more or less in different standards), 3500 common characters covering 99.48% of all characters and 7000 general characters. Therefore, there might be multiple possible characters for each syllable. For example, totally 60 characters in 3500 common characters have the same syllable "yi". The task of Chinese pinyin-to-character conversion is to disambiguate the corresponding character for a syllable with other characters.

The common approach for Chinese pinyin-to-character conversion is n-gram Language Model (LM), which selects the word/character candidate sequence with greatest probability. The determination of word/character for current syllables depends on the probability on previous words/characters (detailed description in next section). There are two major problems for word n-gram language model. One problem is the mistaken recognition of Out-Of-Vocabulary (OOV) words. Since the OOV words are not in segmented training data, they might be given low probabilities in word n-gram LM. The other problem is that n-gram language model only embodies the constraints on nearby words, which lacks long distance constraints in grammatic or syntactic structure.

Many approaches are proposed to deal with these problems. Discriminative methods, such as maximum entropy model (Xiao *et al*., 2007), support vector machines (Jiang *et al*., 2007), can employ pinyin features after current word/character rather than only using information before current word/character in word n-gram LM. Rule-based post-processing approach brings grammatical rule constraints to candidates, where these rules can be obtained by hand-crafted or statistical methods. Wang employed rough set theory to extract rules automatically from training dataset (Wang *et al*., 2004).

In this study, we proposed a linear reranking model for Chinese pinyin-to-character conversion problem. Our model can utility the information from word/character n-gram LMs, character-based discriminative model, pinyin-word occurrence model, Part-Of-Speech (POS) tagging model, word-POS co-occurrence model and dependency model. We can combine these sub models by using their probability outputs for candidates. Minimum error training method is used to obtain the weights for sub models. We perform experiments on Lancaster Corpus of Mandarin to analyze the effect of different sub models on pinyin-to-character conversion and the results show that the information from sub models helps the conversion.

Corresponding Author: Xinxin Li, Computer Application Research Center, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China, Tel.: +86 075526033790

PINYIN-TO-CHARACTER CONVERSION

Given a syllable sequence S = s_1, s_2, …, s_n, the aim of Chinese pinyin-to-character conversion is to find

jian chi gai ge 、 kai fang ，
坚持 改革 、 开放 ，
insist on reform and open

Fig. 1: An example for pinyin-to-character conversion

a sequence of characters C = c_1, c_2, …, c_n, or a sequence of words W = w_1, w_2, …w_n, where each word w_k is composed of one or more characters c_i, …, c_j. An example of Chinese pinyin-to-character conversion is shown in Fig. 1.

Using word n-gram LM, word sequence W for inputting syllable sequence S satisfying:

$$w^* = \arg\max_W P(W \mid S)$$
$$= \arg\max_W P(S \mid W)P(W) / P(S)$$

where,

$$P(S \mid W) = \prod_{i=1}^{m} p(s_i \mid w_i)$$

$$P(W) = \prod_{i=1}^{m} p(w_i \mid w_1, ..., w_{i-1})$$

For trigram language model,

$$P(W) = p(w_1)p(w_2 \mid w_1)\prod_{i=1}^{m} p(w_i \mid w_{i-2}, w_{i-1})$$

Using character n-gram LM, the character sequence C can be determined by:

$$C^* = \arg\max_C P(C \mid S)$$
$$= \arg\max_C P(S \mid C)P(C) / P(S)$$

where, P(S|C) and P(C) are calculated similar as P(S|C) and P(W).

When applying word n-gram LM, a beam search decoding method is used for Chinese pinyin-to-character conversion. Normally, P (S|W) and P (S|C) are omitted in the decoding phase. For each position j in syllable sentence, the decoding algorithm will generate all words in vocabulary according to current syllable sequence s_i, .., s_j ending on position j and calculate the probability $p(w_c|w_{c-2},w_{c-1})$ of current word w_c over previous words.

For every position j, it keeps the n-best word sequences. The best word sequence will be the one with greatest probability on sentence end.

Preliminary experiments: We perform experiments on The Lancaster Corpus of Mandarin Chinese (LCMC) to analysis the performance of word n-gram LM and then compare it with single character n-gram LM, oracle of mixed result with character n-gram LM and oracle of k-best word sequences generated by word n-gram LM. LCMC contains 15 different text categories, with totally 45735 segmented, POS tagged, pinyin annotated sentences. The original pinyin text is annotated using pinyin4j tool which only considers the most probable syllable for each character separately. We re-annotated the pinyin of sentences using minimum word number matching algorithm on Sogou dictionary. The re-annotated pinyin sequence is more accurate than the original one.

Table 1: Statistics of training, development and test data

Dataset	Train	Dev	Test
#Sentences	1135296	8251	8637
#All words	875397	62961	63608
#Chn words	723424	52039	52263
#Chn characters	1144559	82527	82788
#OOV word	13692	1043	1245
OOV rate	1.56%	1.66%	1.96%

Table 2: Performance of different LMs on development data

Models	CER	IVWER	OOWER
character	12.92	11.25	15.04
word	11.27	9.03	14.23
mixed oracle	7.38	6.89	11.58
500-best oracle	2.01	4.01	5.94

To evaluate word n-gram LM, we selected 200 sentences from each category as development and test dataset separately and the rest 39735 sentences as training data. All the training, development and test sentences are first segmented by Chinese end punctuations and then filtered by eliminating those containing English words. The statistics of our training, development and test data are listed in Table 1.

The training data for our word and character n-gram LMs are taken from People's Diary of year 1998, 2006, 2007 and 2009-2012. To obtain the full list of Chinese words is impossible because the specification of word segmentation is not determined and the number of words is still grown. Even all words are given, it's inpractical because an accurate word n-gram LM needs a huge training data and the size of the trained language model is too huge to use. Our word dictionary for word n-gram LM contains 130750 words selected from four sources: the 7000 general characters, the 56064 common words, words from XinHua dictionary and 94412 highest frequency words extracted from Google Chinese web 5g version 1. For character n-gram LM, the word dictionary contains only the 7000 general characters. We use a minimum word number method to segment the raw training text and SRILM tool to train our language model (Stolcke, 2002). The Out-Of-Vocabulary (OOV) words in Table 1 are defined as words that are not contained in word dictionary.

The performance of word and character n-gram LMs are evaluated on development dataset. The results are shown on Table 2. Both models are evaluated on Character Error Rate (CER) measure:

$$CER = \frac{the\ number\ of\ correct\ characters}{the\ number\ of\ all\ characters}$$

Both models are also evaluated by IV word error rate, OOV word error rate. The OOV words are defined on table 1 are Chinese words not in word dictionary. The length of both IV words and OOV words is large than 1. Table 2 shows that word n-gram LM achieves better performance than character n-gram LM in all evaluation measures, providing evidence that word n-gram LM provides more constraints than character n-gram LM. And word n-gram LM achieves better performance on IV words than OOV words.

Oracle of mixed result of word and character n-gram LMs and oracle of k-best word lists are calculated to compare with word n-gram LM. The k-best word lists are generated using beam search algorithm on word-based model. Table 2 shows that the word and character n-gram LMs cause different errors and oracle of their mixed result obtains better performance. The oracle of k-best word sequences achieves the best CER. The fact that the performance of word n-gram LM is much lower than both oracles leaves us a space to improve it by using other information.

LINEAR RERANKING MODEL

Experimental results show single word n-gram LM is insufficient for Chinese pinyin-to-character conversion. We proposed a linear reranking model to combine different information. In this study, several sub models are introduced into our model, including part-of-speech tagging model, pinyin-word co-occurrence model, word-POS occurrence model and dependency model. To utility these different sub models, we can combine their probability outputs of each candidate and use minimum error training method to obtain the weight for these models. The k-best candidates for each pinyin sentence are first generated by word n-gram LM using beam search algorithm and then POS tagged and dependency parsed. Our linear reranking model is used to select the best one from these candidates.

Minimum Error Training (MERT) algorithm is proposed to combine different features for machine translation (Och, 2003). It has been successfully used in joint word segmentation and POS tagging (Jiang *et al*., 2008). The probability of a possible corresponding word sequence W for a syllable sequence S is calculated as:

$$
\begin{aligned}
&P_{mert}(W \mid S) = P_{mert}(W, C, T, D \mid S) \\
&= \sum_{i=0}^{k} w_i * P_{sub}(W, C, T, D \mid S) \\
&= w_0 * P(W) + w_1 * P(C) + w_2 * P(S \mid C) \\
&+ w_3 * P_{occur}(W \mid S) + w_4 * P_{occur}(S \mid W) \\
&+ w_5 * P(T \mid W) + w_6 * P(T) \\
&+ w_7 * P_{occur}(W \mid T) \\
&+ w_8 * P(D \mid W, T)
\end{aligned}
$$

Table 3: Feature templates for character-based model

1	c_n (n = -2..2)
2	$c_n c_{n+1}$ (n = -1..0)
3	$c_{-1} c_1$

where, $\sum_{i=1}^{8} w_i = 1$ and P_{sub}(W, C, T, D|S) represents the probability output of sub models, including word n-gram LM P(W), character n-gram LM P(C), character-based discriminative model P(S|C), pinyin-word co-occurrence model P_{occur}(W|S) and P_{occur}(S|W), POS tagging model P(T|W), POS N-gram LM P(T), POS-word occurrence model P_{occur}(W|T) and a dependency model P(D|W, T). The first line in the equation is valid because a word sequence W corresponds to only one POS sequence T and dependency tree D.

The weights w_j($1 \le j \le k$) can be obtained by keeping other weights fixed and calculating each weight w_j iteratively. The probability of the candidate is:

$$P_{mert}(W \mid S) = w_j * P_j(W \mid S) + \sum_{i \neq j} w_i * P_i(W \mid S)$$

The left probability $w_j * P_j(W|S)$ is an variable, the right probability $\sum_{i \neq j} w_i^* P_i(W|S)$ is an constant.

A direct grid search algorithm for determining each weight is not suitable for the task, since it's expensive to re-calculate the probabilities of all candidates to find the best one. MERT employs a piece-wise linear search algorithm for each jth dimension. The optimum value of each weight w_j must be in the intersection values of all lines depicted by the formula above. It's easy to find the optimum one because when the critical value changes, only a few candidates need to be calculated. The details of MERT algorithm has been described in Zaidan (2009).

Next, we will describe the sub models we employed and the method of calculating the probability output of the candidate for each sub model.

Character-based discriminative model: The character-based model is a discriminative model determining each character in a sentence using the information only from pinyin sequence. The model is trained using averaged perceptron algorithm. The pinyin feature templates are listed in Table 3.

$$
\begin{aligned}
&F(C) = \arg\max_{C \in GEN(P)} P(S \mid C) \\
&= \arg\max_{C \in GEN(P)} \Phi(S, C) \times \bar{\alpha}
\end{aligned}
$$

P(S|C) is the probability output. The features Φ(S, C) can be generated before decoding, so the time for decoding can be enormously reduced. The details of decoding algorithm and parameter estimation method can be found in Collins (2002) and Li *et al*. (2011).

Pinyin-word co-occurrence: For syllables s, there will be multiple possible corresponding words w. For a syllable sequence S and its corresponding word sequence W, we define pinyin-word co-occurrence as:

Table 4: Feature template for POS model

1	$w_{-2}t_0$	End (w_{-1}) w_0 start (w_1) t_0
2	$w_{-1}t_0$	When len (w_0) = 1
3	w_0t_0	Start (w_0) t_0
4	w_1t_0	End (w_0) t_0
5	w_2t_0	c_nt_0, (n = 1, len (w_0-2)
6	$t_{-1}t_0$	Start (w_0) c_nt_0, (n = above)
7	$t_{-2}t_{-1}t_0$	End (w_0) c_nt_0, (n = above)
8	$t_{-1}w_0$	$c_nc_{n+1}t_0$, (c_n= c_{n+1})
9	$w_0t_0$end (w_{-1})	Class (start (w_0)) t_0
10	$w_0t_0$start (w_1)	Class (end (w_0)) t_0

Fig. 2: An example of a dependency tree

$$P_{occur}(W \mid S) = \prod_{i=1}^{m} p(w_i \mid s_i),$$
$$P_{occur}(S \mid W) = \prod_{i=1}^{m} p(s_i \mid w_i)$$

$p(w_i|w_i)$ and $p(s_i|w_i)$ can be calculated using minimum likelihood estimation (MLE) method, where,

$$p(w_i \mid s_i) = \frac{N(w_i, s_i)}{N(s_i)}, p(s_i \mid w_i) = \frac{N(w_i, s_i)}{N(w_i)}$$

We can statistic the number of syllables $N(s_i)$ and syllable-word pairs N(wi, s_i) from annotated dataset. The training dataset is the same as the dataset for word n-gram LM.

POS tagging model: Given the word sequence W, we can determine its Part-Of-Speech (POS) tags. Previous work reveals that POS information improves word segmentation (Ng and Low, 2004; Zhang and Nivre, 2011). In this study, we introduce POS tagging model trying to help selecting the best word sequence. Averaged perceptron algorithm is used for parameter estimation.

The POS tag sequence T for word sequence W is chosen as:

$$P(T \mid W) = \arg\max_{T \in GEN(w)} P(T \mid W)$$
$$= \arg\max_{T \in GEN(W)} \Phi(T, W) \times \bar{\alpha}$$

The features Φ (T, W) for POS tagging model are defined similar as Zhang and Clark (2008), shown in Table 4.

The training data for POS tagging is taken from Chinese Treebank (CTB5) and the distribution of training, development and test dataset is same as Zhang and Clark (2008). The F-1 measure of our POS model on golden segmented development dataset is 95.26%.

POS N-gram language model: The POS n-gram LM is trained using the data described before and POS tagged. We define the probability of POS sequence T as:

$$P(T) = \prod_{i=1}^{m} p(t_i \mid t_1, ..., t_{i-1})$$

POS-word co-occurrence: The POS-word Co-occurrence is defined similar as syllable-word co-occurrence. For word sequence w and its corresponding POS sequence T, the co-occurrence is defined as:

$$P_{occur}(W \mid T) = \prod_{i=1}^{m} p(w_i \mid t_i),$$
$$P_{occur}(T \mid W) = \prod_{i=1}^{m} p(t_i \mid w_i)$$

where, $p(w_i|t_i)$ and $p(t_i|w_i)$ are also calculated using MLE method.

Dependency model: For a long sentence W, the determination of a word for syllables might not be determined only using the information from nearby words and syllables. It might also need grammatic and syntactic information. For example, in a syllable sequence "yi zhi mei li ke ai de xiao hua mao". The determination of word for syllables "yi zhi" is dependent on its relation with syllables "xiao hua mao". Dependency model brings long-distance word relation for Chinese pinyin-to-character conversion. For a segmented and POS tagged sentence (W, T), we use a deterministic transition-based algorithm for dependency parsing (Zhang and Nivre, 2011). Figure 2 gives an example of a dependency tree structure.
The probability of dependency tree D is:

$$P(D \mid W, T) = \prod_i w_i \times f_i(D \mid W, T)$$

EXPERIMENTS AND ANALYSIS

The experimental dataset for linear reranking model is the same as preliminary experiments on word n-gram LM in section 2. Same as calculating oracle of k best word sequences, we take 500 candidate dependency trees for each syllable sequence. Then linear reranking model is used to select the best one from these candidate ones. Character Error Rate (CER) is used as our evaluation measure.

Experiments on sub models: All our sub models can generate a probability for a give dependency tree D, as we listed in section (Linear Reranking Model). We train these sub models on training dataset and evaluate the linear reranking model on development dataset.

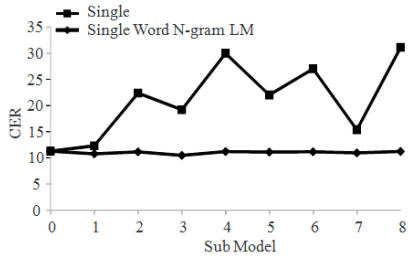

Fig. 3: Performance of sub models on development data

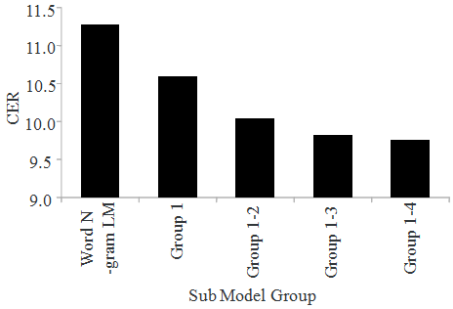

Fig. 4: Performance of sub model group increase on development data

Table 5: Experimental results of linear reranking model on development data

Seq	Sub Models	CER
A	All	9.76
A-0	All-word N-gram LM	10.47
A-1	All-character N-gram LM	10.10
A-2	All-character-based model	9.83
A-3	All-pinyin-word occurrence	10.06
A-4	All-POS model	9.84
A-5	All-word-POS occurrence	9.83
A-6	All-POS-Word occurrence	9.76
A-7	All-POS N-gram LM	9.78
A-8	All-dependency model	9.82

Table 6: Experimental results of sub model groups on development data

Group	Sub models	CER
Character	0,1,2	10.59
Pinyin	0,3,4	10.33
POS	0,5,6,7	10.84
Dependency	0,8	11.21

We first employ a backward greedy search algorithm to find the suitable sub model sets. The procedure starts with the linear model with all sub models and evaluates it on development data. First, it will iteratively remove each sub model and re-evaluate the performance. Then, the linear model will remove the sub model with greatest improvement on development data. The procedure will continue until the performance is not improved.

The results are shown in Table 5. After using the sub model selection strategy, word-pinyin occurrence $P_{occur}(T|W)$ is eliminated because it doesn't the improve the performance of the linear model. We compare the linear reranking model of all rest sub models and the models with removing one sub model each time.

Table 5 shows that all the sub models have positive impact for the task, in which word n-gram LM improves the performance most, character N-gram LM ranks second and pinyin-word occurrence ranks third. The dependency model also brings an improvement on development data, better than POS-word occurrence model.

We then evaluate the impact of each sub model on the task. The experimental results are shown in Fig. 3. The numbers 0, .., 8 in X axis denote sub models listed in Table 5. The above curve in Fig. 3 describes the results of only one sub model used for evaluation. The results reveal similar phenomenon as Table 5 that word n-gram LM achieves the lowest CER and character n-gram LM ranked the second. The dependency model gives the lowest accuracy.

The below curve shows the results of linear reranking models of word n-gram LM combining with one other sub models. The linear reranking models outperform all single sub models. Given word n-gram LM, adding pinyin-word occurrence model achieves the best performance where it's omitted in the original decoding algorithm described in (Pinyin-to-Character Conversion).

Experiments on sub model group: We then evaluate different groups of sub models to validate which information has the most influent on the conversion. All sub models are split into four groups: character group, pinyin group, POS group, dependency group. The results in Table 6 show that except word n-gram model, the pinyin group provides the most useful information and the dependency model gives the less benefit.

Figure 4 shows experimental results of the linear reranking models beginning from word n-gram LM and adding one sub model group each time. The entire linear reranking model benefits from every sub model group. The final model achieves 1.51 points decrease than word N-gram LM, about 14.49% increase on performance.

Complexity analysis: The space and time complexity of the linear reranking model is vital for Chinese pinyin-to-character conversion. All sub models can be generated along with character appending in sentence with linear time complexity of sentence length O (n). Then our linear reranking model is also linear with sentence length since it's a linear combination of these sub models.

Experiments on test data: We then evaluate our models on test data. The experimental results exhibit similar performance as development data. With more sub models added, the performance of linear reranking model increases. The final model achieves 10.94 CER, about 8.89% decrease than 12.03 CER of word n-gram LM.

CONCLUSION

In this study, we employ a linear reranking model to solve Chinese pinyin-character conversion problem. Our reranking model can improve the performance of word n-gram LM by utilizing information from different sub models. There are two directions we can continue to work on. One is extending our model to speech recognition. The second direction is to reduce the size of sub models while keeping its accuracy.

REFERENCES

Collins, M., 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, pp: 1-8.

Jiang, W., G. Guan, X. Wang and B. Liu, 2007. Pinyin to character conversion model based on support vector machines. J. Chinese Inf. Proces., 21(2): 100-105.

Jiang, W., L. Huang, Q. Liu and Y. Lü, 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. Proceedings of ACL-08: HLT, Columbus, Ohio, pp: 897-904.

Li, X., W. Wang and L. Yao, 2011. Joint decoding for Chinese word segmentation and pos tagging using character-based and word-based discriminative models. 2011 International Conference on Asian Language Processing (IALP), Penang, Malaysia, pp: 11-14.

Ng, H.T. and J.K. Low, 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? Word-based or character-based? In: Lin, D. and D. Wu (Eds.), Proceedings of EMNLP 2004, Barcelona, Spain, pp: 277-284.

Och, F.J., 2003. Minimum error rate training in statistical machine translation. Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, pp: 160-167.

Stolcke, A., 2002. Srilm - an extensible language modeling toolkit. Proceedings of the International Conference on Spoken Language Processing, Denver, Colorado, pp: 901-904.

Wang, X., Q. Chen and D.S. Yeung, 2004. Mining pinyin-to-character conversion rules from large-scale corpus: A rough set approach. IEEE T. Syst. Man Cy, B, 34(2): 834-844.

Xiao, J., B. Liu and X. Wang, 2007. Exploiting pinyin constraints in pinyin-to-character conversion task: A class-based maximum entropy markov model approach. Comput. Linguist. Chinese Language Proces., 12(3): 325-348.

Zaidan, O., 2009. Z-mert: A fully configurable open source tool for minimum error rate training of machine translation systems. Prague Bull. Math. Linguistics, 91(1): 79-88.

Zhang, Y. and S. Clark, 2008. Joint word segmentation and pos tagging using a single perceptron. Proceedings of ACL-08: HLT, Columbus, Ohio, pp: 888-896.

Zhang, Y. and J. Nivre, 2011. Transition-based dependency parsing with rich non-local features. Proceedings of the 49th Annual Meeting of the Association for Computational Linguis-tics: Human Language Technologies, Portland, Oregon, USA, pp: 188-193.