A Noisy Channel Model for Grapheme-based Machine Transliteration

Yuxiang Jia, Danqing Zhu, Shiwen Yu
Institute of Computational Linguistics, Peking University, Beijing, China
Key Laboratory of Computational Linguistics, Ministry of Education, China
{yxjia,zhudanqing,yusw}@pku.edu.cn

Abstract

Machine transliteration is an important Natural Language Processing task. This paper proposes a Noisy Channel Model for Grapheme-based machine transliteration. Moses, a phrase-based Statistical Machine Translation tool, is employed for the implementation of the system. Experiments are carried out on the NEWS 2009 Machine Transliteration Shared Task English-Chinese track. English-Chinese back transliteration is studied as well.

1 Introduction

Transliteration is defined as phonetic translation of names across languages. Transliteration of Named Entities is necessary in many applications, such as machine translation, corpus alignment, cross-language information retrieval, information extraction and automatic lexicon acquisition.

The transliteration modeling approaches can be classified into phoneme-based, grapheme-based and hybrid approach of phoneme and grapheme.

Many previous studies are devoted to the phoneme-based approach (Knight and Graehl, 1998; Virga and Khudanpur, 2003). Suppose that *E* is an English name and *C* is its Chinese transliteration. The phoneme-based approach first converts *E* into an intermediate phonemic representation *p*, and then converts *p* into its Chinese counterpart *C*. The idea is to transform both source and target names into comparable phonemes so that the phonetic similarity between two names can be measured easily.

The grapheme-based approach has also attracted much attention (Li et al., 2004). It treats the transliteration as a statistical machine translation problem under monotonic constraint. The idea is to obtain the bilingual orthographical correspondence directly to reduce the possible errors introduced in multiple conversions.

The hybrid approach attempts to utilize both phoneme and grapheme information for transliteration. (Oh and Choi, 2006) proposed a way to fuse both phoneme and grapheme features into a single learning process.

The rest of this paper is organized as follows. Section 2 briefly describes the noisy channel model for machine transliteration. Section 3 introduces the model's implementation details. Experiments and analysis are given in section 4. Conclusions and future work are discussed in section 5.

2 Noisy Channel Model

Machine transliteration can be regarded as a noisy channel problem. Take the English-Chinese transliteration as an example. An English name *E* is considered as the output of the noisy channel with its Chinese transliteration *C* as the input. The transliteration process is as follows. The language model generates a Chinese name *C*, and the transliteration model converts *C* into its back-transliteration *E*. The channel decoder is used to find $\hat{C}$ that is the most likely to the word *C* that gives rise to *E*. $\hat{C}$ is the result transliteration of *E*.

The process can be formulated with equation 1.

$$\hat{C} = \arg\max_{C} P(C \mid E) = \arg\max_{C} \frac{P(C)P(E \mid C)}{P(E)} \quad (1)$$

Since *P(E)* is constant for the given *E*, we can rewrite equation 1 as follows:

$$\hat{C} = \arg\max_{C} P(C)P(E \mid C) \quad (2)$$

The language model *P(C)* is simplified as n-gram model of Chinese characters and is trained with a Chinese name corpus. The transliteration model *P(E|C)* is estimated from a parallel corpus of English names and their Chinese transliterations. The channel decoder combines the lan-

Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 88–91,
Suntec, Singapore, 7 August 2009. ©2009 ACL and AFNLP

guage model and transliteration model to generate Chinese transliterations for given English names.

3 Implementation

Moses (Koehn et al., 2007), a phrase-based statistical machine translation tool, is leveraged to implement the noisy channel model for grapheme-based machine transliteration without reordering process (Matthews, 2007). Figure 1 is an illustration of the phrase alignment result in machine transliteration of the name pairs “Clinton” and “克林顿”, where characters are as words and combinations of characters are as phrases.

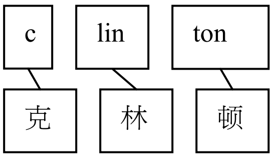

Figure 1. Example phrase alignment

A collection of tools are used by Moses. SRILM is used to build statistical language models. GIZA++ is used to perform word alignments over parallel corpora. Mert is used for weight optimization. It includes several improvements to the basic training method including randomized initial conditions and permuted model order and dynamic parameter range expansion or restriction. Bleu, an automatic machine translation evaluation metric, is used during Mert optimization. Moses’ beam-search decoding algorithm is an efficient search algorithm that quickly finds the highest probability translation among the exponential number of choices.

Moses automatically trains translation models for any language pairs with only a collection of parallel corpora. The parallel transliteration corpora need to be preprocessed at first. English names need to be lowercased. Both English names and Chinese transliterations are space delimited. Samples of preprocessed input are shown in figure 2.

a a b y e	奥 比
a a g a a r d	埃 格 德
a a l l i b o n e	阿 利 本
a a l t o	阿 尔 托
a a m o d t	阿 莫 特

Figure 2. Sample preprocessed name pairs

4 Experiments

This section describes the data sets, experimental setup, experiment results and analysis.

4.1 Data Sets

The training set contains 31961 paired names between English and Chinese. The development set has 2896 pairs. 2896 English names are given to test the English-Chinese transliteration performance.

Some statistics on the training data are shown in table 1. All the English-Chinese transliteration pairs are distinct. English names are unique while some English names may share the same Chinese transliteration. So the total number of unique Chinese names is less than that of English names. The Chinese characters composing the Chinese transliterations are limited, where there are only 370 unique characters in the 25033 Chinese names. Supposing that the name length is computed as the number of characters it contains, the average length of English names is about twice that of Chinese names. Name length is useful when considering the order of the character n-gram language model.

#unique transliteration pairs	31961
#unique English names	31961
#unique Chinese names	25033
#unique Chinese characters	370
Average number of English characters per name	6.8231
Average number of Chinese characters per name	3.1665
Maximum number of English characters per name	15
Maximum number of Chinese characters per name	7

Table 1. Training data statistics

4.2 Experimental setup

Both English-Chinese forward transliteration and back transliteration are studied. The process can be divided into four steps: language model building, transliteration model training, weight tuning, and decoding. When building language model, data smoothing techniques Kneser-Ney and interpolate are employed. In transliteration model training step, the alignment heuristic is grow-diag-final, while other parameters are default settings. Tuning parameters are all defaults. When decoding, the parameter distortion-limit is set to 0, meaning that no reordering operation is needed. The system outputs the 10-best distinct transliterations.

The whole training set is used for language model building and transliteration model training. The development set is used for weight tuning and system testing.

4.3 Evaluation Metrics

The following 6 metrics are used to measure the quality of the transliteration results (Li et al., 2009a): Word Accuracy in Top-1 (ACC), Fuzziness in Top-1 (Mean F-score), Mean Reciprocal Rank (MRR), MAP_{ref}, MAP_{10}, and MAP_{sys}.

In the data of English-Chinese transliteration track, each source name only has one reference transliteration. Systems are required to output the 10-best unique transliterations for every source name. Thus, MAP_{ref} equals ACC, and MAP_{sys} is the same or very close to MAP_{10}. So we only choose ACC, Mean F-score, MRR, and MAP_{10} to show the system performance.

4.4 Results

The language model n-gram order is an important factor impacting transliteration performance, so we experiment on both forward and back transliteration tasks with increasing n-gram order, trying to find the order giving the best performance. Here the development set is used for testing.

Figure 3 and 4 show the results of forward and back transliteration respectively, where the performances become steady when the order reaches 6 and 11. The orders with the best performance in all metrics for forward and back transliteration are 2 and 5, which may relate to the average length of Chinese and English names.

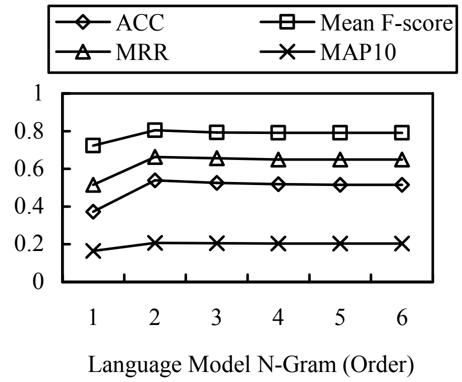

Figure 3. E2C language model n-gram (forward)

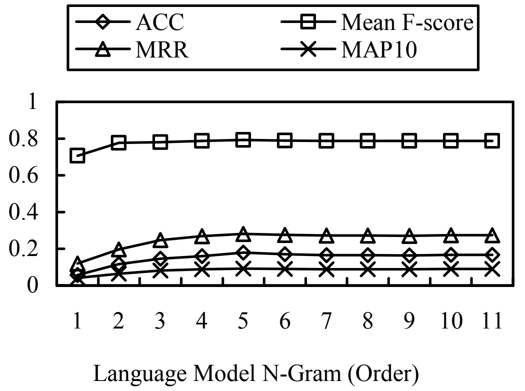

Figure 4. E2C language model n-gram (back)

Weights generated in the training step can be optimized through the tuning process. The development set, 2896 name pairs, is divided into 4 equal parts, 1 for testing and other 3 for tuning. We take the best settings as the baseline, and increase tuning size by 1 part at one time. Table 2 and 3 show the tuning results of forward and back transliteration, where the best results are boldfaced. Tuning set size of 0 refers to the best settings before tuning. Performances get improved after tuning, among which the ACC of forward transliteration gets improved by over 11%. The forward transliteration performance gets improved steadily with the increase of tuning set size, while the back transliteration performance peaks at tuning set size of 2.

Tuning size	ACC	Mean F-score	MRR	MAP_{10}
0	0.543	0.797	0.669	0.209
1	0.645	0.851	0.752	0.231
2	0.645	0.850	0.749	0.230
3	**0.655**	**0.854**	**0.758**	**0.233**

Table 2. E2C tuning performance (forward)

Tuning size	ACC	Mean F-score	MRR	MAP_{10}
0	0.166	0.790	0.278	0.092
1	0.181	0.801	0.306	0.102
2	**0.190**	**0.806**	**0.314**	**0.104**
3	0.187	0.801	0.312	0.104

Table 3. E2C tuning performance (back)

Table 2 shows that forward transliteration performance gets improved with the increase of tuning set size, so we use the whole development set as the tuning set to tune the final system and the final official results from the shared task report (Li et al., 2009b) are shown in table 4.

ACC	Mean F-score	MRR	MAP_{ref}	MAP_{10}	MAP_{sys}
0.652	0.858	0.755	0.652	0.232	0.232

Table 4. The final official results of E2C forward

Experiments show that forward transliteration has better performance than back transliteration. One reason may be that on average English name is longer than Chinese name, thus need more data to train a good character level language model. Another reason is that some information is lost during transliteration which can not be recovered in back transliteration. One more very important reason is as follows. Typically in back transliteration, you have only one correct reference transliteration, and therefore, a wide coverage word level language model is very useful. Without it, back transliteration may have a poor performance.

5 Conclusions and future work

This paper proposes a Noisy Channel Model for grapheme-based machine transliteration. The phrase-based statistical machine translation tool, Moses, is leveraged for system implementation. We participate in the NEWS 2009 Machine Transliteration Shared Task English-Chinese track. English-Chinese back transliteration is also studied. This model is language independent and can be applied to transliteration of any language pairs.

To improve system performance, extensive error analyses will be made in the future and methods will be proposed according to different error types. We will pay much attention to back transliteration for its seemingly greater difficulty and explore relations between forward and back transliteration to seek a strategy solving the two simultaneously.

Acknowledgements

The authors are grateful to the organizers of the NEWS 2009 Machine Transliteration Shared Task for their hard work to provide such a good research platform. The work in this paper is supported by a grant from the National Basic Research Program of China (No.2004CB318102) and a grant from the National Natural Science Foundation of China (No.60773173).

References

K. Knight and J. Graehl. 1998. Machine Transliteration. Computational Linguistics, Vol. 24, No. 4, pp. 599-612.

P. Virga and S. Khudanpur. 2003. Transliteration of Proper Names in Cross-lingual Information Retrieval. In Proceedings of the ACL Workshop on Multi-lingual Named Entity Recognition 2003.

H.Z. Li, M. Zhang and J. Su. 2004. A Joint Source Channel Model for Machine Transliteration. In Proceedings of the 42nd ACL, pp. 159-166.

J.H. Oh and K.S. Choi. 2006. An Ensemble of Transliteration Models for Information Retrieval. In Information Processing and Management, Vol. 42, pp. 980-1002.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin and E. Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In Proceedings of the 45th ACL Companion Volume of the Demo and Poster Sessions, pp. 177-180.

D. Matthews. 2007. Machine Transliteration of Proper Names. Master thesis. University of Edinburgh.

H.Z. Li, A. Kumaran, M. Zhang and V. Pervouchine. 2009a. Whitepaper of NEWS 2009 Machine Transliteration Shared Task. In Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009), Singapore.

H.Z. Li, A. Kumaran, V. Pervouchine and M. Zhang. 2009b. Report on NEWS 2009 Machine Transliteration Shared Task. In Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009), Singapore.