Reranking with Multiple Features for Better Transliteration

Yan Song[†] **Chunyu Kit**[†] **Hai Zhao**[‡†]
[†]Department of Chinese, Translation and Linguistics
City University of Hong Kong, 83 Tat Chee Ave., Kowloon, Hong Kong
[‡]Department of Computer Science and Engineering
Shanghai Jiao Tong University, #800, Dongchuan Rd, Shanghai, China
{yansong,ctckit}@cityu.edu.hk, zhaohai@cs.sjtu.edu.cn

Abstract

Effective transliteration of proper names via grapheme conversion needs to find transliteration patterns in training data, and then generate optimized candidates for testing samples accordingly. However, the top-1 accuracy for the generated candidates cannot be good if the right one is not ranked at the top. To tackle this issue, we propose to rerank the output candidates for a better order using the averaged perceptron with multiple features. This paper describes our recent work in this direction for our participation in NEWS2010 transliteration evaluation. The official results confirm its effectiveness in English-Chinese bidirectional transliteration.

1 Introduction

Since transliteration can be considered a direct orthographic mapping process, one may adopt general statistical machine translation (SMT) procedures for its implementation. Aimed at finding phonetic equivalence in another language for a given named entity, however, different transliteration options with different syllabification may generate multiple choices with the symphonic form for the same source text. Consequently, even the overall results by SMT output are acceptable, it is still unreliable to rank the candidates simply by their statistical translation scores for the purpose of selecting the best one. In order to make a proper choice, the direct orthographic mapping requires a precise alignment and a better transliteration option selection. Thus, powerful algorithms for effective use of the parallel data is indispensable, especially when the available data is limited in volume.

Interestingly, although an SMT based approach could not achieve a precise top-1 transliteration result, it is found in (Song et al., 2009) that, in contrast to the ordinary top-1 accuracy (ACC) score, its recall rate, which is defined in terms of whether the correct answer is generated in the n-best output list, is rather high. This observation suggests that if we could rearrange those outputs into a better order, especially, push the correct one to the top, the overall performance could be enhanced significantly, without any further refinement of the original generation process. This reranking strategy is proved to be efficient in transliteration generation with a multi-engine approach (Oh et al., 2009).

In this paper, we present our recent work on reranking the transliteration candidates via an online discriminative learning framework, namely, the averaged perceptron. Multiple features are incorporated into it for performance enhancement. The following sections will give the technical details of our method and present its results for NEWS2010 shared task for named entity transliteration.

2 Generation

For the generation of transliteration candidates, we follow the work (Song et al., 2009), using a phrase-based SMT procedure with the log-linear model

$$P(t|s) = \frac{exp[\sum_{i=1}^{n} \lambda_i h_i(s,t)]}{\sum_t exp[\sum_{i=1}^{n} \lambda_i h_i(s,t)]} \qquad (1)$$

for decoding. Originally we use two directional phrase[1] tables, which are learned for both directions of source-to-target and target-to-source, containing different entries of transliteration options. In order to facilitate the decoding by exploiting all possible choices in a better way, we combine the forward and backward directed phrase tables together, and recalculate the probability for each en-

[1]It herein refers to a character sequence as described in (Song et al., 2009).

try in it. After that, we use a phoneme resource[2] to refine the phrase table by filtering out the wrongly extracted phrases and cleaning up the noise in it. In the decoding process, a dynamic pruning is performed when generating the hypothesis in each step, in which the threshold is variable according to the current searching space, for we need to obtain a good candidate list as precise as possible for the next stage. The parameter for each feature function in log-linear model is optimized by MERT training (Och, 2003). Finally, a maximum number of 50 candidates are generated for each source name.

3 Reranking

3.1 Learning Framework

For reranking training and prediction, we adopt the averaged perceptron (Collins, 2002) as our learning framework, which has a more stable performance than the non-averaged version. It is presented in Algorithm 1. Where $\vec{\omega}$ is the vector of parameters we want to optimize, x, y are the corresponding source (with different syllabification) and target graphemes in the candidate list, and Φ represents the feature vector in the pair of x and y. In this algorithm, reference y_i^* is the most appropriate output in the candidate list according to the true target named entity in the training data. We use the Mean-F score to identify which candidate can be the reference, by locating the one with the maximum Mean-F score value. This process updates the parameters of the feature vector and also relocate all of the candidates according to the ranking scores, which are calculated in terms of the resulted parameters in each round of training as well as in the testing process. The number of iteration for the final model is determined by the development data.

3.2 Multiple Features

The following features are used in our reranking process:

Transliteration correspondence feature, $f(s_i, t_i)$;

This feature describes the mapping between source and target graphemes, similar to the transliteration options in the phrase table in our previous generation process, where s and t refer to the source and target language respectively, and i to the current position.

[2]In this work, we use Pinyin as the phonetic representation for Chinese.

Algorithm 1 Averaged perceptron training

```
Input: Candidate list with reference
    ${LIST(x_j, y_j)_{j=1}^{n}, y_i^*}_{i=1}^{N}$
Output: Averaged parameters
 1: $\vec{\omega} \leftarrow 0, \vec{\omega}_a \leftarrow 0, c \leftarrow 1$
 2: for $t = 1$ to $T$ do
 3:     for $i = 1$ to $N$ do
 4:         $\hat{y}_i \leftarrow argmax_{y \in LIST(x_j, y_j)} \vec{\omega} \cdot \Phi(x_i, y_i)$
 5:         if $\hat{y}_i \neq y_i^*$ then
 6:             $\vec{\omega} \leftarrow \vec{\omega} + \Phi(x_i^*, y_i^*) - \Phi(\hat{x}_i, \hat{y}_i)$
 7:             $\vec{\omega}_a \leftarrow \vec{\omega}_a + c \cdot \{\Phi(x_i^*, y_i^*) - \Phi(\hat{x}_i, \hat{y}_i)\}$
 8:         end if
 9:         $c \leftarrow c + 1$
10:     end for
11: end for
12: return $\vec{\omega} - \vec{\omega}_a / c$
```

Source grapheme chain feature, $f(s_{i-1}^{i})$;

It measures the syllabification for a given source text. There are two types of units in different levels. One is on syllable level, e.g., "*aa/bye*", "*aa/gaar/d*", reflecting the segmentation of the source text, and the other on character level, such as "*a/b*", "*a/g*", "*r/d*", showing the combination power of several characters. These features on different source grapheme levels can help the system to achieve a more reliable syllabification result from the candidates. We only consider bi-grams when using this feature.

Target grapheme chain feature, $f(t_{i-2}^{i})$;

This feature measures the appropriateness of the generated target graphemes on both character and syllables level. It performs in a similar way as the language model for SMT decoding. We use *tri*-gram syllables in this learning framework.

Paired source-to-target transition feature, $f(<s, t>_{i-1}^{i})$;

This type of feature is firstly proposed in (Li et al., 2004), aiming at generating source and target graphemes simultaneously under a suitable constraint. We use this feature to restrict the synchronous transition of both source and target graphemes, measuring how well are those transitions, such as for "*st*", whether "*s*" transliterated by "斯" is followed by "*t*" transliterated by "特". In order to deal with the data sparseness, only bi-gram transition relations are considered in this feature.

Hidden Markov model (HMM) style features;

There are a group of features with HMM style constraint for evaluating the candidates generated in previous SMT process, including, previous syllable HMM features, $f(s_{i-n+1}^{i}, t_i)$, posterior syllable HMM features, $f(s_{i}^{i+n-1}, t_i)$, and posterior character HMM features, $f(s_i, l, t_i)$, where l denotes the character following the previous syllable in the source language. For the last feature, it is effective to use both the current syllable and the first letter of the next syllable to bound the current target grapheme. The reason for applying this feature in our learning framework is that, empirically, the letters following many syllables strongly affect the transliteration for them, e.g., *Aves* $\rightarrow$ 埃维斯, "*a*" followed by "*v*" is always translated into "埃" rather than "阿".

Target grapheme position feature, $f(t_i, p)$;

This feature is an improved version of that proposed in (Song et al., 2009), where p refers to the position of t_i. We have a measure for the target graphemes according to their source graphemes and the current position of their correspondent target characters. There are three categories of such position, namely, start (S), mediate (M) and end (E). S refers to the first character in a target name, E to the final, and the others belong to M. This feature is used to exploit the observation that some characters are more likely to appear at certain positions in the target name. Some are always found at the beginning of a named entity while others only at the middle or the end. For example, "*re*" associated to first character in a target name is always transliterated as "雷", such as *Redd* $\rightarrow$ 雷德. When "*re*" appears at the end of a source name, however, its transliteration will be "尔" in most cases, just like *Gore* $\rightarrow$ 戈尔.

Target tone feature;

This feature is only applied to the transliteration task with Chinese as the target language. It can be seen as a combination of a target grapheme chain with some position features, using tone instead of the target grapheme itself for evaluation. There are 5 tones (0,1,2,3,4) for Chinese characters. It is easy to conduct a comprehensive analysis for the use of a higher ordered transition chain as a better constraint. Many fixed tone patterns can be identified in the Chinese transliteration training data. The tone information can also be extracted from the Pinyin resource we used in the previous stage.

Besides the above string features, we also have some numeric features, as listed below.

Transliteration score;

This score is the joint probabilities of all transliteration options, included in the output candidates generated by our decoder.

Target language model score;

This score is calculated from the probabilistic tri-gram language model.

Source/target Pinyin feature;

This feature uses Pinyin representation for a source or target name, depending on what side the Chinese language is used. It measures how good the output candidates can be in terms of the comparison between English text and Pinyin representation. The resulted score is updated according to the Levenshtein distance for the two input letter strings of English and Pinyin.

For a task with English as the target language, we add the following two additional features into the learning framework.

Vowel feature;

It is noticed that when English is the target language, vowels can sometimes be missing in the generated candidates. This feature is thus used to punish those outputs unqualified to be a valid English word for carrying no vowel.

Syllable consistent feature;

This feature measures whether an English target name generated in the previous step has the same number of syllables as the source name. In Chinese-to-English transliteration, Chinese characters are single-syllabled, thus

Table 1: Evaluation results for our NEWS2010 task.

Task	Source	Target	ACC	Mean F	MRR	Map_ref	Recall	ACC$_{SMT}$
EnCh	English	Chinese	0.477	0.740	0.506	0.455	0.561	0.381
ChEn	Chinese	English	0.227	0.749	0.269	0.226	0.371	0.152

we can easily identify their number. For syllabification, we have an independent segmentation process for calculating the syllables.

4 Results

For NEWS2010, we participated in all two Chinese related transliteration tasks, namely, EnCh (English-to-Chinese) and ChEn (Chinese-to-English back transliteration). The official evaluation scores for our submissions are presented in Table 1 with recall rate, and the ACC score (ACC$_{SMT}$) for original SMT outputs. It is easy to see the performance gain for the reranking, and also from the recall rate that there is still some room for improvement, in spite of the high ratio of ACC/Recall[3] calculated from Table 1. However, it is also worth noting that, some of the source texts cannot be correctly transliterated, due to many multiple-word name entities with semantic components in the test data, e.g., "*MANCHESTER BRIDGE*", "*BRIGHAM CITY*" etc. These semantic parts are beyond our transliteration system's capability to tackle, especially when the training data is limited and the only focus of the system is on the phonetic equivalent correspondence.

Compared to the EnCh transliteration, we get a rather low ACC score for the ChEn back transliteration, suggesting that ChEn task is somewhat harder than the EnCh (in which Chinese characters are always limited). The ChEn task is a one-to-many translation, involving a lot of possible choices and combinations of English syllables. This certainly makes it a more challengeable task than EnCh. However, looking into the details of the outputs, we find that, in the ChEn back transliteration, some characters in the test corpus are unseen in the training and the development data, resulting in incorrect transliterations for many graphemes. This is another factor affecting our final results for the ChEn task.

5 Conclusion

In this paper, we have presented our work on multiple feature based reranking for transliteration generation. It NEWS2010 results show that this approach is effective and promising, in the sense that it ranks the best in EnCh and ChEn tasks. The reranking used in this work can also be considered a regeneration process based on an existing set, as part of our features are always used directly to generate the initial transliteration output in other researches. Though, those features are strongly dependent on the nature of English and Chinese languages, it is thus not an easy task to transplant this model for other language pairs. It is an interesting job to turn it into a language independent model that can be applied to other languages.

[3]Compared to the results from (Song et al., 2009)

Acknowledgments

The research described in this paper was partially supported by City University of Hong Kong through the Strategic Research Grants (SRG) 7002267 and 7008003. Dr. Hai Zhao was supported by the Natural Science Foundation of China (NSFC) through the grant 60903119. We also thank Mr. Wenbin Jiang for his helpful suggestions on averaged perceptron learning.

References

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP-2002*, pages 1–8, July.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of ACL-04*, pages 159–166, Barcelona, Spain, July.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL-03*, pages 160–167, Sapporo, Japan, July.

Jong-Hoon Oh, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Machine transliteration using target-language grapheme and phoneme: Multi-engine transliteration approach. In *Proceedings of NEWS 2009*, pages 36–39, Suntec, Singapore, August.

Yan Song, Chunyu Kit, and Xiao Chen. 2009. Transliteration of name entity via improved statistical translation on character sequences. In *Proceedings of NEWS 2009*, pages 57–60, Suntec, Singapore, August.