Chinese-English Backward Transliteration Assisted with Mining Monolingual Web Pages

Fan Yang, Jun Zhao, Bo Zou, Kang Liu, Feifan Liu
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
{fyang,jzhao,bzou,kliu,ffliu}@nlpr.ia.ac.cn

Abstract

In this paper, we present a novel backward transliteration approach which can further assist the existing statistical model by mining monolingual web resources. Firstly, we employ the syllable-based search to revise the transliteration candidates from the statistical model. By mapping all of them into existing words, we can filter or correct some pseudo candidates and improve the overall recall. Secondly, an AdaBoost model is used to rerank the revised candidates based on the information extracted from monolingual web pages. To get a better precision during the reranking process, a variety of web-based information is exploited to adjust the ranking score, so that some candidates which are less possible to be transliteration names will be assigned with lower ranks. The experimental results show that the proposed framework can significantly outperform the baseline transliteration system in both precision and recall.

1 Introduction[*]

The task of Name Entity (NE) translation is to translate a name entity from source language to target language, which plays an important role in machine translation and cross-language information retrieval (CLIR). Transliteration is a subtask in NE translation, which translates NEs based on the phonetic similarity. In NE translation, most person names are transliterated, and some parts of location names or organization names also need to be transliterated. Transliteration has two directions: forward transliteration which transforms an original name into target language, and backward transliteration which recovers a name back to its original expression. For instance, the original English person name "Clinton" can be forward transliterated to its Chinese expression "克/ke 林/lin顿/dun" and the backward transliteration is the inverse processing. In this paper, we focus on backward transliteration from Chinese to English.

Many previous researches have tried to build a transliteration model using statistical approach [Knight and Graehl, 1998; Lin and Chen, 2002; Virga and Khudanpur, 2003; Gao, 2004]. There are two main challenges in statistical backward transliteration: First, statistical transliteration approach selects the most probable translations based on the knowledge learned from the training data. This approach, however, does not work well when there are multiple standards [Gao, 2004]. Second, backward transliteration is more challenging than forward transliteration as it is required to disambiguate the noises introduced in the forward transliteration and estimate the original name as close as possible [Lin and Chen, 2002]. One of the most important causes in introducing noises is that: some silent syllables in original names have been missing when they are transliterated to target language. For example, when "Campbell" is transliterated into "坎/kan贝/bei尔/er", the "p" is missing.

In order to make up the disadvantages of statistical approach, some researchers have been seeking for the assistance of web resource. [Wang et al., 2004; Cheng et al., 2004; Nagata et al., 2001; Zhang et al, 2005] used bilingual web pages to extract translation pairs. Other efforts have been made to combine a statistical transliteration model with web mining [Al-Onaizan and Knight, 2002; Long Jiang et al, 2007]. Most of these methods need bilingual resources. However, those kinds of resources are not readily available in many cases. Moreover, to search for bilingual pages, we have to depend on the performance of search engines. We can't get Chinese-English bilingual pages when the input is a Chinese query. Therefore, the existing

[*]Contact: Jun ZHAO, jzhao@nlpr.ia.ac.cn.

assistance approaches using web-mining to assist transliteration are not suitable for Chinese to English backward transliteration.

Thus in this paper, we mainly focus on the following two problems to be solved in transliteration.

Problem I: Some silent syllables are missing in English-Chinese forward transliteration. How to recover them effectively and efficiently in backward transliteration is still an open problem.

Problem II: Statistical transliteration always chooses the translations based on probabilities. However, in some cases, the correct translation may have lower probability. Therefore, more studies are needed on combination with other techniques as supplements.

Aiming at these two problems, we propose a method which mines monolingual web resources to assist backward transliteration. The main ideas are as follows. We assume that for every Chinese entity name which needs to be backward transliterated to an English original name, the correct transliteration exists somewhere in the web. What we need to do is to find out the answers based on the clues given by statistical transliteration results. Different from the traditional methods which extract transliteration pairs from bilingual pages, we only use monolingual web resources. Our method has two advantages. Firstly, there are much more monolingual web resources available to be used. Secondly, our method can revise the transliteration candidates to the existing words before the subsequent re-ranking process, so that we can better mine the correct transliteration from the Web.

Concretely, there are two phases involved in our approach. In the first phase, we split the result of transliteration into syllables, and then a syllable-based searching processing can be employed to revise the result in a word list generated from web pages, with an expectation of higher recall of transliteration. In the second phase, we use a revised word as a search query to get its contexts and hit information, which are integrated into the AdaBoost classifier to determine whether the word is a transliteration name or not with a confidence score. This phase can readjust the candidate's score to a more reasonable point so that precision of transliteration can be improved. Table 1 illustrates how to transliterate the Chinese name "阿/a加/jia 西/xi" back to "Agassi".

Chinese name	Transliteration results	Revised Candidate	Re-rank Results
阿加西 a jia xi Agassi	aggasi agahi agacy agasie …	agasi agathi agathe **agassi** …	**agassi** agasi agache agga …

Table 1. An example of transliteration flow

The experimental results show that our approach improves the recall from 41.73% to 59.28% in open test when returning the top-100 results, and the top-5 precision is improved from 19.69% to 52.19%.

The remainder of the paper is structured as follows. Section 2 presents the framework of our system. We discuss the details of our statistical transliteration model in Section 3. In Section 4, we introduce the approach of revising and re-ranking the results of transliteration. The experiments are reported in Section 5. The last section gives the conclusion and the prediction of future work.

2 System Framework

Our system has three main modules.

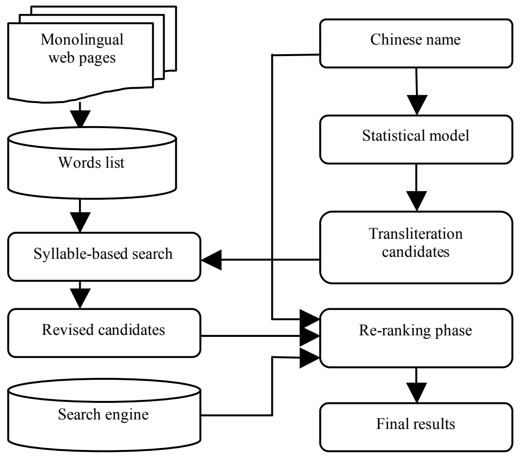

Figure 1. System framework

1) ***Statistical transliteration***: This module receives a Chinese Pinyin sequence as its input, and output the *N*-best results as the transliteration candidates.

2) ***Candidate transliteration revision through syllable-based searching***: In the module, a transliteration candidate is transformed into a syllable query. We use a syllable-based searching strategy to select the revised candidate from a huge word list. Each word in the list is indexed by syllables, and the similarity between the word and the query is calculated. The most similar words are returned as the revision results. This module guar-

antees the transliteration candidates are all existing words.

3) ***Revised candidate re-ranking in web pages***: In the module, we search the revised candidates to get their contexts and hit information which we can use to score the probability of being a transliteration name. This phase doesn't generate new candidates, but re-rank the revised candidate set to improve the performance in top-5.

Under this framework, we can solve the two problems of statistical model mentioned above.

(1) The silent syllables will be given lower weights in syllable-based search, so the missing syllables will be recovered through selecting the most similar existing words which can contain some silent syllables.

(2) The query expansion technology can recall more potential transliteration candidates by expanding syllables to their "synonymies". So the mistakes introduced when selecting syllables in statistical transliteration will be corrected through giving suitable weights to synonymies.

Through the revision phase, the results of statistical model which may have illegal spelling will be mapped to its most similar existing words. That can improve the recall. In re-ranking phase, the revised candidate set will be re-ranked to put the right answer on the top using hybrid information got from web resources. So the precision of transliteration will be improved.

3 Statistical Transliteration Model

We use syllables as translation units to build a statistical Chinese-English backward transliteration model in our system.

3.1 Traditional Statistical Translation Model

[P. Brown et al., 1993] proposed an IBM source-channel model for statistical machine translation (SMT). When the channel output $f=f_1,f_2 \ldots. f_n$ observed, we use formula (1) to seek for the original sentence $e=e_1,e_2 \ldots. e_n$ with the most likely posteriori.

$$e' = \arg\max_e P(e \mid f) = \arg\max_e P(f \mid e)P(e) \qquad (1)$$

The translation model $P(f \mid e)$ is estimated from a paired corpus of foreign-language sentences and their English translations. The language model $P(e)$ is trained from English texts.

3.2 Our Transliteration Model

The alignment method is the base of statistical transliteration model. There are mainly two kinds of alignment methods: phoneme-based alignment [Knight and Graehl, 1998; Virga and Khudanpur, 2003] and grapheme-based alignment [Long Jiang, 2007]. In our system, we adopt the syllable-based alignment from Chinese pinyin to English syllables, where the syllabication rules mentioned in [Long Jiang et al., 2007] are used.

For example, Chinese name "希/xi 尔/er 顿/dun" and its backward transliteration "Hilton" can be aligned as follows. "Hilton" is split into syllable sequence as "hi/l/ton", and the alignment pairs are "xi-hi", "er-l", "dun-ton".

Based on the above alignment method, we can get our statistical Chinese-English backward transliteration model as,

$$E = \arg\max_E p(PY \mid ES)p(ES) \qquad (2)$$

Where, PY is a Chinese Pinyin sequence, ES is a English syllables sequence, $p(PY \mid ES)$ is the probability of translating ES into PY, $p(ES)$ is the generative probability of a English syllable language model.

3.3 The Difference between Backward Transliteration and Traditional Translation

Chinese-English backward transliteration has some differences from traditional translation.

1) We don't need to adjust the order of syllables when transliteration.

2) The language model in backward transliteration describes the relationship of syllables in words. It can't work as well as the language model describing the word relationship in sentences.

We think that the crucial problem in backward transliteration is selecting the right syllables at every step. It's very hard to obtain the exact answer only based on the statistical transliteration model. We will try to improve the statistical model performance with the assistance of mining web resources.

4 Mining Monolingual Web Pages to Assist Backward Transliteration

In order to get assistance from monolingual Web resource to improve statistical transliteration, our method contains two main phases: "revision" and "re-ranking". In the revision phase, transliteration candidates are revised using syllable-based search in the word list, which are generated by collecting the existing words in web pages. Because the process of named entity recognition may lose some NEs, we will reserve all the words in web corpus without any filtering. The revision process can improve the recall through correcting some mistakes in the transliteration results of statistical model.

In the re-ranking phase, we search every revised candidate on English pages, score them according to their contexts and hit information so that the right answer will be given a higher rank.

4.1 Using Syllable-based Retrieval to Revise Transliteration Candidates

In this section, we will propose two methods respectively for the two problems of statistical model mentioned in section 1.

4.1.1 Syllable-based retrieval model

When we search a transliteration candidate tc_i in the word list, we firstly split it into syllables $\{es_1,es_2,\ldots..es_n\}$. Then this syllable sequence is used as a query for syllable-based searching.

We define some notions here.

- Term set $\boldsymbol{T}=\{t_1,t_2\ldots.t_k\}$ is an orderly set of all syllables which can be viewed as terms.
- Pinyin set $\boldsymbol{P}=\{py_1,py_2\ldots.py_k\}$ is an orderly set of all Pinyin.
- An input word can be represented by a vector of syllables $\{es_1,es_2,\ldots..es_n\}$.

We calculate the similarity between a transliteration result and each word in the list to select the most similar words as the revised candidates. The $\{es_1,es_2,\ldots..,es_n\}$ will be transformed into a vector $V_{\text{query}}=\{t_1,t_2\ldots.t_k\}$ where t_i represents the ith term in $\boldsymbol{T}$. The value of t_i is equal to 0 if the ith term doesn't appear in query. In the same way, the word in list can also be transformed into vector representation. So the similarity can be calculated as the inner product between these two vectors.

We don't use *tf* and *idf* conceptions as traditional information retrieval (IR) to calculate the terms' weight. We use the weight of t_i to express the expectation probability of ith term having pronunciation. If the term has a lower probability of having pronunciation, its weight is low. So when we searching, the missing silent syllables in the results of statistical transliteration model can be recovered because such syllables have little impact on similarity measurement. The formula we used is as follows.

$$Sim(query, word) = \frac{V_{query} \times V_{word}}{L_{word} / L_{py}} \tag{3}$$

The numerator is the inner product of two vectors. The denominator is the length of word L_{word} divided by the length of Chinese pinyin sequence L_{py}. In this formula, the more syllables in one word, the higher score of inner production it may get, but the word will get a loss for its longer length. The word which has the shortest length and the highest syllable hitting ratio will be the best.

Another difference from traditional IR is how to deal with the order of the words in a query. According to transliteration, the similarity must be calculated under the limitation of keeping order, which can't be satisfied by current methods. We use the algorithm like calculating the edit distance between two words. The syllables are viewed as the units which construct a word. The edit distance calculation finds the best matching with the least operation cost to change one word to another word by using deletion/addition/insertion operations on syllables. But the complexity will be too high to afford if we calculate the edit distance between a query and each word in the list. So, we just calculate the edit distance for the words which get high score without the order limitation. This trade off method can save much time but still keep performance.

4.1.2 Mining the Equivalent through Syllable Expansion

In most collections, the same concept may be referred to using different words. This issue, known as synonymy, has an impact on the recall of most information retrieval systems. In this section, we try to use the expansion technology to solve problem II. There are three kinds of expansions to be explained below.

Syllable expansion based on phonetic similarity: The syllables which correspond to the same Chinese pinyin can be viewed as synonymies. For example, the English syllables "din" and "tin" can be aligned to the same Chinese pinyin "ding".

Given a Chinese pinyin sequence $\{py_1,py_2,\ldots.py_n\}$ as the input of transliteration model, for every py_i, there are a set of syllables

$\{es_1, es_2 \ldots\ldots es_k\}$ which can be selected as its translation. The statistical model will select the most probable one, while others containing the right answer are discarded. To solve this problem, we expand the query to take the synonymies of terms into consideration. We create an expansion set for each Chinese pinyin. A syllable es_i will be selected into the expansion set of py_j based on the alignment probability $P(es_i|py_j)$ which can be extracted from the training corpus. The phonetic similarity expansion is based on the input Chinese Pinyin sequence, so it's same for all candidates.

Syllable expansion based on syllable similarity: If two syllables have similar alignment probability with every pinyin, we can view these two syllables as synonymy. Therefore, if a syllable is in the query, its synonymies should be contained too. For example, "fea" and "fe" can replace each other.

To calculate the similarity, we first obtain the alignment probability $P(py_j|es_k)$ of every syllable. Then the distance between any two syllables will be calculated using formula (4).

$$Sim(es_j, es_k) = \frac{1}{N}\sum_{i=1}^{N} P(py_i \mid es_j)P(py_i \mid es_k) \quad (4)$$

This formula is used to evaluate the similarity of two syllables in alignment. The expansion set of the *i*th syllable can be generated by selecting the most similar N syllables. This kind of expansion is conducted upon the output of statistical transliteration model.

Syllable expansion based on syllable edit distance: The disadvantage of last two expansions is that they are entirely dependent on the training set. In other word, if some syllables haven't appeared in the training corpus, they will not be expanded. To solve the problem, we use the method of expansion based on edit distance. We use edit distance to measure the similarity between two syllables, one is in training set and the other is absent. Because the edit distance expansion is not very relevant to pronunciation, we will give this expansion method a low weight in combination. It works when new syllables arise.

Combine the above three strategies: We will combine the three kinds of expansion method together. We use the linear interpolation to integrate them. The formulas are follows.

$$S = (1-\alpha)S_{pre} + \alpha S_{sy} + \beta S_{ed} \quad (5)$$

$$S = (1-\alpha)S_{pre} + \alpha S_{py} + \beta S_{ed} \quad (6)$$

where S_{pre} is the score of exact matching, S_{sy} is the score of expansion based on syllables similarity and S_{py} based on phonetic similarity. We will adjust these parameters to get the best performance. The experimental results and analysis will be reported in section 5.3.

4.2 Re-Ranking the Revised Candidates Set using the Monolingual Web Resource

In the first phase, we have generated the revised candidate set $\{rc_1, rc_2, \ldots, rc_n\}$ from the word list using the transliteration results as clues. The objective is to improve the overall recall. In the second phase, we try to improve the precision, i.e. we wish to re-rank the candidate set so that the correct answer will be put in a higher rank.

[Al-Onaizan et al., 2002] has proposed some methods to re-score the transliteration candidates. The limitation of their approach is that some candidates are propbale not existing words, with which we will not get any information from web. So it can only re-rank the transliteration results to improve the precision of top-5. In our work, we can improve the recall of transliteration through the revising process before re-ranking.

In this section, we employ the AdaBoost framework which integrates several kinds of features to re-rank the revised candidate set. The function of the AdaBoost classifier is to calculate the probability of the candidate being a NE. Then we can re-rank the revised candidate set based on the score. The features used in our system are as follows.

NE or not: Using rc_i as query to search for monolingual English Web Pages, we can get the context set $\{T_{i1}, T_{i2} \ldots\ldots T_{in}\}$ of rc_i. Then for every T_{ik}, we use the named entity recognition (NER) software to determine whether rc_i is a NE or not. If rc_i is recognized as a NE in some T_{ik}, rc_i will get a score. If rc_i can't be recognized as NE in any contexts, it will be pruned.

The hit of the revised candidate: We can get the hit information of rc_i from search engine. It is used to evaluate the importance of rc_i. Unlike [Al-Onaizan et al., 2002], in which the hit can be used to eliminate the translation results which contain illegal spelling, we just use hit number as a feature.

The limitation of compound NEs: When transliterating a compound NE, we always split them into several parts, and then combine their transliteration results together. But in this circumstance, every part can add a limitation in the selection of the whole NE. For example: “希/xi拉/la里/li · 克/ke林/lin顿/dun” is a compound name. “希/xi拉/la里/li” can be transliterate to “Hilary” or “Hilaly” and “克/ke林/lin顿/dun” can be transliterate to “Clinton” or “Klinton”. But the combination of “Hilary·Clinton” will be selected for it is the most common combination. So the hit of combination query will be extracted as a feature in classifier.

Hint words around the NE: We can take some hint words around the NE into the query, in order to add some limitations to filter out noisy words. For example: “总统 (president)” can be used as hint word for “克林顿 (Clinton)”. To find the hint words, we first search the Chinese name in Chinese web pages. The frequent words can be extracted as hint words and they will be translated to English using a bilingual dictionary. These hint words are combined with the revised candidates to search English web pages. So, the hit of the query will be extracted as feature.

The formula of AdaBoost is as follow.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{7}$$

Where α_t is the weight for the *i*th weak classifier $h_t(x)$. α_t can be calculated based on the precision of its corresponding classifier.

5 Experiments

We carry out experiments to investigate how much the revision process and the re-ranking process can improve the performance compared with the baseline of statistical transliteration model. We will also evaluate to which extents we can solve the two problems mentioned in section 1 with the assistance of Web resources.

5.1 Experimental data

The training corpus for statistical transliteration model comes from the corpus of Chinese <-> English Name Entity Lists v 1.0 (LDC2005T34). It contains 565,935 transliteration pairs. Ruling out those pairs which are not suitable for the research on Chinese-English backward transliteration, such as Chinese-Japanese, we select a training set which contains 14,443 pairs of Chinese-European & American person names. In the training set, 1,344 pairs are selected randomly as the close test data. 1,294 pairs out of training set are selected as the open test data. To set up the word list, a 2GB-sized collection of web pages is used. Since 7.42% of the names in the test data don’t appear in the list, we use *Google* to get the web page containing the absent names and add these pages into the collection. The word list contains 672,533 words.

5.2 Revision phase vs. statistical approach

Using the results generated from statistical model as baseline, we evaluate the revision module in recall first. The statistical transliteration model works in the following 4 steps: 1) Chinese name are transformed into pinyin representation and the English names are split into syllables. 2) The *GIZA++*[1] tool is invoked to align pinyin to syllables, and the alignment probabilities $P(py \mid es)$ are obtained. 3) Those frequent sequences of syllables are combined as phrases. For example, “be/r/g”$\rightarrow$”berg”, “s/ky”$\rightarrow$”sky”. 4) *Camel*[2] decoder is executed to generate 100-best candidates for every name.

We compare the statistical transliteration results with the revised results in Table 2. From Table 2 we can find that the recall of top-100 after revision is improved by 13.26% in close test set and 17.55% in open test set. It proves that the revision module is effective for correcting the mistakes made in statistical transliteration model.

	Transliteration results		Revised results	
	close	open	close	open
Top1	33.64%	9.41%	27.15%	11.04%
Top5	40.37%	13.38%	42.83%	19.69%
Top10	47.79%	17.56%	56.98%	26.52%
Top20	61.88%	25.44%	71.05%	37.81%
Top50	66.49%	36.19%	82.16%	46.22%
Top100	**72.52%**	**41.73%**	**85.78%**	**59.28%**

Table 2. Statistical model vs. Revision module

To show the effects of the revision on the two above-mentioned problems in which the statistical model does not solve well: the losing of silent syllables and the selection bias problem, we make a statistics of the improvements with a measurement of “correction time”.

For a Chinese word whose correct transliteration appears in top-100 candidates only if it has been

[1] http://www.fjoch.com/GIZA++.html
[2] http://www.nlp.org.cn

revised, we count the "correction time". For example, when "Argahi" is revised to "Agassi" the correction time is "1" for Problem II and "1" for Problem I, because in "hi"→ "si" the syllable is expanded, and in "si" →"ssi" an "s" is added.

	Close test	Open test
Problem I	0.6931	0.7853
Problem II	0.9264	1.1672

Table 3. Average time of correction

This measurement reflects the efficiency of the revision of search strategy, in contrast to those spelling correction techniques in which several operations of "add" and "expand" are inevitable. It has proved that the more an average correction time is, the more efficient our strategy is.

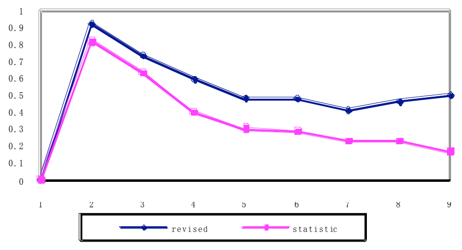

Figure 2. Length influence in recall comparison

The recall of the statistical model relies on the length of English name in some degree. It is more difficult to obtain an absolutely correct answer for longer names, because they may contain more silent and confused syllables. However, through the revision phase, this tendency can be effectively alleviated. In Figure 2, we make a comparison between the results of the statistical model and the revision module with the changing of syllable's length in open test. The curves demonstrate that the revision indeed prevents the decrease of recall for longer names.

5.3 Parameter setting in the revision phase

We will show the experimental results when setting different parameters for query expansion. In the expansion based on phonetic similarity, for every Chinese pinyin, we select at most 20 syllables to create an expansion set. We set $\beta = 0.1$ in formula (5). The results are shown in the columns labeled "exp1" in Table 4.

From the results we can conclude that, we get the best performance when $\alpha = 0.4$. That means the performance is best when the weight of exact matching is a little larger than the weight of fuzzy matching. We can also see that, higher weight of exact matching will lead to low recall, while higher weight of fuzzy matching will bring noise in.

The expansion method based on syllable similarity is also evaluated. For every syllable, we select at most 15 syllables to create the expansion set. We set $\beta = 0.1$. The results are shown in the columns labeled "exp2" in Table 4.

From the results we can conclude that, we get the best performance when $\alpha = 0.5$. It means that we can't put emphasis on any matching methods. Comparison with the expansion based on phonetic similarity, the performance is poorer. It means that the expansion based on phonetic similarity is more suitable for revising transliteration candidates.

5.4 Revision phase vs. re-ranking phase

After the phase of revising transliteration candidates, we re-rank the revised candidate set with the assistance of monolingual web resources. In this section, we will show the improvement in precision after re-ranking.

We have selected four kinds of features to integrate in the AdaBoost framework. To determine whether the candidate is NE or not in its context, we use the software tool *Lingpipe*^{3}. The queries are sent to *google*, so that we can get the hit of queries and the top-10 snippets will be extracted as context.

The comparison of revision results and re-ranking results is shown as follows.

	Revised results		Re-ranked results	
	close	open	close	open
Top1	27.15%	11.04%	**58.08%**	**38.63%**
Top5	42.83%	19.69%	**76.35%**	**52.19%**
Top10	56.98%	26.52%	83..92%	54.33%
Top20	71.05%	37.81%	83.92%	57.61%
Top50	82.16%	46.22%	83.92%	57.61%
Top100	**85.78%**	**59.28%**	85.78%	59.28%

Table 5. Revision results vs. Re-ranking results

From these results we can conclude that, after re-ranking phase, the noisy words will get a lower

^{3} http://www.alias-i.com/lingpipe/

	$\alpha = 0.2$		$\alpha = 0.3$		$\alpha = 0.4$		$\alpha = 0.5$		$\alpha = 0.6$		$\alpha = 0.7$		$\alpha = 0.8$	
	exp1	exp2	exp1	exp2	exp1	exp2	exp1	exp2	exp1	exp2	exp1	exp2	exp1	exp2
Top1	13.46	13.32	13.79	13.61	**11.04**	12.70	11.65	**10.93**	10.83	11.25	9.62	10.63	8.73	10.18
Top5	21.58	19.59	23.27	20.17	**19.69**	18.28	21.07	**17.25**	22.05	16.84	17.90	16.26	17.38	15.34
Top10	27.39	22.71	28.41	24.73	**26.52**	22.93	26.83	**21.81**	27.26	20.39	24.38	21.20	25.42	18.20
Top20	35.23	34.88	35.94	29.49	**37.81**	31.57	38.59	**33.04**	36.52	31.72	35.25	29.75	34.65	27.62
Top50	43.91	40.63	43.75	40.85	**46.22**	41.46	48.72	**42.79**	45.48	40.49	41.57	39.94	42.81	38.07
Top100	53.76	48.47	54.38	52.04	**59.28**	53.15	57.36	**53.46**	55.19	51.83	55.63	49.52	53.41	47.15

Table 4. Parameters Experiment

rank. Through the revision module, we get both higher recall and higher precision than statistical transliteration model when at most 5 results are returned.

We also use the average rank and average reciprocal rank (ARR) [Voorhees and Tice, 2000] to evaluate the improvement. ARR is calculated as

$$ARR = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{R(i)} \tag{8}$$

where $R(i)$ is the rank of the answer of ith test word. M is the size of test set. The higher of ARR, the better the performance is.

The results are shown as Table 6.

	Statistical model		Revision module		Re-rank Module	
	close	open	close	open	close	open
Average rank	37.63	70.94	24.52	58.09	16.71	43.87
ARR	0.3815	0.1206	0.3783	0.1648	0.6519	0.4492

Table 6. ARR and AR evaluation

The ARR after revision phase is lower than the statistical model. Because the goal of revision module is to improve the recall as possible as we can, some noisy words will be introduced in. The noisy words will be pruned in re-ranking module. That is why we get the highest ARR value at last. So we can conclude that the revision module improves recall and re-ranking module improves precision, which help us get a better performance than pure statistical transliteration model

6 Conclusion

In this paper, we present a new approach which can revise the results generated from statistical transliteration model with the assistance of monolingual web resource. Through the revision process, the recall of transliteration results has been improved from 72.52% to 85.78% in the close test set and from 41.73% to 59.28% in open test set, respectively. We improve the precision in re-ranking phase, the top-5 precision can be improved to 76.35% in close test and 52.19% in open test. The promising results show that our approach works pretty well in the task of backward transliteration.

In the future, we will try to improve the similarity measurement in the revision phase. And we also wish to develop a new approach using the transliteration candidates to search for their right answer more directly and effectively.

Acknowledgments

The work is supported by the National High Technology Development 863 Program of China under Grants no. 2006AA01Z144, the National Natural Science Foundation of China under Grants No. 60673042, the Natural Science Foundation of Beijing under Grants no. 4073043.

References

Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In Proc.of ACL-02.

Kevin Knight and Jonathan Graehl. 1998. Machine Transliteration. Computational Linguistics 24(4).

Wei-Hao Lin and Hsin-His Chen. 2002 Backward Machine Transliteration by Learning Phonetic Similarity. In Proc. Of the 6th CoNLL

Donghui Feng, Yajuan Lv, and Ming Zhou. 2004. A New Approach for English-Chinese Named Entity Alignment. In Proc. of EMNLP-2004.

Long Jiang, Ming Zhou, Lee-Feng Chien, and Cheng Niu, 2007. Named Entity Translation with Web Mining and Transliteration. In Proc. of IJCAI-2007.

Wei Gao. 2004. Phoneme-based Statistical Transliteration of Foreign Name for OOV Problem. A thesis of Master. The Chinese University of Hong Kong.

Ying Zhang, Fei Huang, Stephan Vogel. 2005. Mining translations of OOV terms from the web through cross-lingual query expansion. SIGIR 2005.

Pu-Jen Cheng, Wen-Hsiang Lu, Jer-Wen Teng, and Lee-Feng Chien. 2004 Creating Multilingual Translation Lexicons with Regional Variations Using Web Corpora. In Proc. of ACL-04

Masaaki Nagata, Teruka Saito, and Kenji Suzuki. 2001. Using the Web as a Bilingual Dictionary. In Proc. of ACL 2001 Workshop on Data-driven Methods in Machine Translation.

Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of proper names in cross-lingual information retrieval. In Proc. of the ACL workshop on Multilingual Named Entity Recognition.

Jenq-Haur Wang, Jei-Wen Teng, Pu-Jen Cheng, Wen-Hsiang Lu, Lee-Feng Chien. 2004. Translating unknown cross-lingual queries in digital libraries using a web-based approach. In Proc. of JCDL 2004.

E.M.Voorhees and D.M.Tice. 2000. The trec-8 question answering track report. In Eighth Text Retrieval Conference (TREC-8)