Phoneme-based Transliteration of Foreign Names for OOV Problem

Wei Gao **Kam-Fai Wong** **Wai Lam**
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{wgao,kfwong,wlam}@se.cuhk.edu.hk

Abstract

One problem seriously affecting CLIR performance is the processing of queries with embedded foreign names. A proper noun dictionary is never complete rendering name translation from English to Chinese ineffective. One way to solve this problem is not to rely on a dictionary alone but to adopt automatic translation according to pronunciation similarities, i.e. to map phonemes comprising an English name to sound units (e.g. pinyin) of the corresponding Chinese name. This process is called transliteration. We present a statistical transliteration method for CLIR applications. An efficient algorithm for phoneme alignment is described. Unlike traditional rule-based approaches, our method is data-driven. So it is independent of dialect features in Chinese. In addition, it is different from other statistical approaches based on source-channel framework in that we adopt a direct transliteration model, i.e. the direction of probabilistic estimation is consistent with transliteration direction. We demonstrate comparable performance on accuracy to other systems.

1 Introduction

Today, it is popular that English Internet surfers search for Chinese documents, and vice versa. This is a typical example of cross language information retrieval (CLIR), in which query expressed in a source language is used to retrieve information represented in another target language. Names of people, places, companies, etc., i.e. proper nouns, are by far the most frequent targets in queries. Contemporary dictionary-based translation techniques are ineffective for proper noun translation as name dictionaries can never be comprehensive. New foreign names appear almost daily; and they become unregistered vocabulary in the dictionary. This brings about the classical Out Of Vocabulary (OOV) problem in lexicography. OOV names could seriously affect translation as well as retrieval performance.

Based on pronunciation similarities, foreign names can usually be translated, or more appropriately transliterated. Transliteration are classified into two directions: Given a word pair *(o, t)* where *o* is the original word in one language and *t* is the transliterated word of *o* in another language. Forward transliteration is the process of phonetic conversion of *o* to *t*; and backward transliteration is generating possible translation candidates and determining the correct *o* given *t* (Lin and Chen, 2002). Transliteration rules are practically mapping templates between the phonemes of target and source names. Existing rule bases are compiled manually; thus, they are not easy to expand and are mostly non-universal, i.e. they are subjected to the interpretation of individual producers. Consider a query searching for Chinese information about "President Bush". The query will be translated to "布殊" for searches in Hong Kong, "布什" for Mainland China and "布希" for Taiwan. In the opposite direction, these Chinese queries should end up with the same English translation. Various dialectical properties in Chinese render rule-based transliteration approach inefficient.

To produce adequate transliterations independent of different dialect features, we propose a data-driven technique for transliterating English names to their Chinese counterparts, i.e. forward transliteration. With the same set of statistics and algorithms, transformation knowledge can be acquired automatically by machine learning from existing origin-transliteration name pairs, irrespective of specific dialectal features implied. Other statistical methods are mostly confined in traditional source-channel framework, in which symbol-mapping probabilities have to be estimated in an inverted direction, our method starts off with direct estimation for transliteration model, which is then combined with target language model for post-processing of generated transliterations.

The rest of this paper is organized as follows: Section 2 summarizes related work, in particular with source-channel model; Section 3 presents our direct transliteration method in detail; Section 4 illustrates how to apply the model in training; Section 5 gives experimental results and analysis; Finally, we conclude this paper.

2 Related Work

The efforts on dealing with forward transliteration consist of how to render phonemes of an original language with phonemes of the target language, and to incorporate target phonemes into combinatorial regularities of the language.

Wan and Verspoor (1998) developed a fully handcrafted rule-based transliteration system for English-Chinese proper nouns. The system first syllabified English words based on a rule set and instance learning. The syllabification module identified syllable boundaries based on consonant clusters and vowels. A sub-syllabification procedure then divided each identified syllable into the form of consonant-vowel, i.e. conforming to the monosyllabic nature of Chinese. The sub-syllables are then mapped to the pinyin equivalents by means of a handcrafted mapping table. This approach is adhoc and dialect dependant.

Meng et al. (2001) presented a learning algorithm for transliteration of OOV names from English to Chinese in the context of Cross-Language Spoken Document Retrieval. They first used a set of handcrafted phonological rules by adding or dropping proper phonemes to normalize English syllables into consonant-vowel format. This aimed to overcome some of the phonological differences between the two languages. The process of cross-lingual phonetic mapping (CLPM) then applied a set of automatically generated transformation rules to map English phonemes to their Chinese counterparts. These rules were learned from aligned parallel data using transformation-based error-driven learning (TEL). However, the manually enumerated rules initiated for CLPM are unable to balance all discrepancies and may introduce errors.

Virga and Khudanpur (2003) demonstrated a fully data-driven transliteration technique for English-Chinese CLIR based on IBM source-channel model, which was originally proposed for French-to-English statistical machine translation (Brown et al., 1993). Their fundamental equation is derived from Bayes' rule:

$$\hat{E} = \arg\max_{E} p(E \mid F) = \arg\max_{E} p(F \mid E) p(E) \quad \text{(Eq-1)}$$

where $F = f_1 f_2 ... f_J$ denotes a *J*-word French sentence as the observation on channel output, and $E = e_1 e_2 ... e_I$ represents *F*'s *I*-word English translation considered as the source of channel input. Given a *F*, the translation process is to find the most probable English sentence $\hat{E}$ so as to indirectly maximize posterior probability *p(E|F)* via optimally combining *inverted*-translation model *p(F|E)* and target language model *p(E)*. When applied to the task of English-to-Chinese name transliteration, the roles of sentence and word were shortened to word and phoneme respectively. Hence, *F* represents a given sequence of *English* phonemes, and *E* a sequence of romanized *Chinese* symbols, i.e. pinyin. *p(F|E)* was trained from sets of name pairs in each language represented by International Phonetic Alphabet (IPA) symbols at English side and pinyin notations at Chinese side. It proceeded in terms of a standard bootstrapping scheme of IBM's translation model training: numbers of EM iterations of Model-1 followed by Model-2, HMM and Model-4. Note that the translation model *p(F|E)* is in an opposite direction, i.e. from Chinese to English. In fact, this is within the same framework as the generative model for Japanese-to-English backward transliteration proposed by Knight and Graehl (1997).

3 Our Forward Transliteration Model

3.1 Why not Source-Channel Model?

Here we rewrite the Eq-1 as follows:

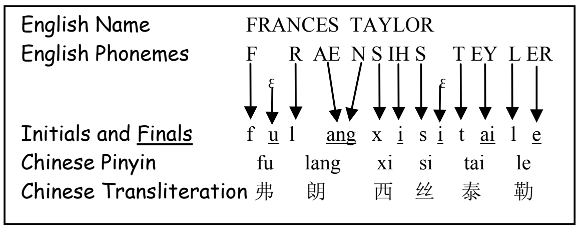
English Name FRANCES TAYLOR
English Phonemes F R AE N S IH S T EY L ER
ε
ε
Initials and Finals f u l ang x i s i t ai l e
Chinese Pinyin fu lang xi si tai le
Chinese Transliteration 弗 朗 西 丝 泰 勒

Figure 1. English-to-Chinese transliteration Example (Virga and Khudanpur, 2003)

$$\widetilde{C} = \arg\max_{C} p(C \mid E) \qquad \text{(Eq-2)}$$

$$\widetilde{C} = \arg\max_{C} \; p(E \mid C)\, p(C) \qquad \text{(Eq-3)}$$

where $E = e_1^{|E|} = e_1 e_2 ... e_{|E|}$ is a sequence of given English phonemes to be transliterated, and $C = c_1^{|C|} = c_1 c_2 ... c_{|C|}$ is the sequence of target Chinese pinyin symbols. $|E|$ and $|C|$ are the number of sound units they contain. Eq-2 is a direct model and Eq-3 is source-channel model. Most of statistical-based transliteration systems, regardless of forward or backward direction, can be classified into the framework of Eq-3.

Basically, Eq-3 is preferable over Eq-2 with the argument that *p(C|E)* has to concentrates its probability as much as possible on well-formed Chinese pinyin strings, whereas inverted translation model *p(E|C)* need not to concentrate on well-formed English phoneme strings because it cooperates with *p(C)*, which is large for well-formed pinyin strings (Brown et al., 1993). However, source-channel model has several limitations:

1. It's hard to extend the baseline of transliteration model by introducing additional dependencies (parameters) (Och and Ney, 2002), such as neighboring phoneme features;

2. Mapping *fertility* constraint: It allows only one target language word to be associated with a contiguous group of source language words, but not vice versa (Papineni et al., 1998). That is, in our task, one English phoneme can never be converted to group of Chinese phonemes. It is a fairly unrealistic restriction. The only example of name pair in (Virga and Khudanpur, 2003) can expose this limitation: /u/[1] and the second /i/ in the third line of Figure 1 are considered as "spurious" sounds produced from dumb sound ε. Under IBM's model, such "inserted" symbols are called having zero *fertility*, which renders the model parameters complicated and sticky. In fact, this results from the difficulties for the *inverted* translation model to consider contiguous pinyin phonemes. It would be more reasonable and easy to handle if /f-u/ (or /s-i/) was looked as common initial-final clusters converted from single English phoneme /F/ (or /S/). This is naturally feasible under our direct model.

[1] Lowercase letters in '/ /' denote Chinese pinyin. Capital letters are English phonemes represented by computer-readable IPA notations – ARPABET.

3.2 Soundness of Direct Model

Eq-2 is the general form of our problem. More specifically, we would like to substitute *p(C|E)* for *p(E|C)* in Eq-3 so that we have a different forward transliteration method as follows:

$$\widetilde{C} = \arg\max_{C} \; p(C \mid E)\, p(C) \qquad \text{(Eq-4)}$$

Our basic idea is that a precise *direct* transliteration model *p(C|E)* concentrates on producing the most likely transcriptions, probably being ill-formed pinyin strings though, for a given E, and the language model *p(C)* helps to make corrections on forms, e.g. eliminating illegal pinyin strings and adjust their rankings. Eq-4 is beyond the Bayes' theorem. Fortunately, we found its mathematical proof under the more general Maximum Entropy principle (Och and Ney, 2002).

Source-channel model failed to estimate the transcription probabilities with respect to some important initial-final clusters in pinyin, which were mapped from single English. The correction for erroneous result sequences was ad hoc and error prone, in which trial candidate finals were inserted between consecutive initials until a legitimate pinyin sequence obtains. In direct model, however, we summarize 4 possible general conditions mapping an English phoneme to pinyin to get rid of this ad hoc process:

1. It maps to an initial or a final;
2. It maps to an initial-final cluster;
3. It maps to dumb sound ε, e.g. /AA K L AH N D/ (Auckland) to /ao ke lan/ (奥克兰), where /D/ is omitted;
4. Insert additional pinyin syllables, e.g. /F L OY D/ (Floyd) to /fo luo yi de/ (佛洛伊德), where /yi/ is an insertion in order to cater for the sound /OY/ which has already been mapped to /uo/.

4 Direct Model Training

4.1 Preprocessing Based on Alignment Templates

The two components in Eq-4, i.e. $p(C|E)$ and $p(C)$, can be trained independently. We first define alignment template between the pair of two sound sequences, E and C. Within total 39 phonemes (24 consonants, 15 vowels) in the English sound inventory and 58 pinyin symbols (23 initials and 35 finals) in Chinese, there are always some indicative sound units (indicators) that are very helpful for alignment. For E, they are:

- All the consonants;
- Vowel at the first position;
- The second vowel of two contiguous vowels.

In C, correspondingly, they are:

- All the Initials;
- Final at the first position;
- The second final of two contiguous finals.

Note similar indicators can be easily identified in other dialectical systems in Chinese as well, such as Cantonese. We define variables as follows:

$\tau(S)$ = # of indicators in sequence S.

$t = max\{\tau(E), \tau(C)\}$

$d = | \tau(E) - \tau(C)|$

We chunk E and C by tagging their indicators and compensate the one with fewer indicators by inserting d dumb sound ε at $min\{\tau(E), \tau(C)\}$ possible positions (ahead of its indicators.). ε is also considered as indicator for participating alignment. So they have identical number of indicators hereafter. The t chunks separated by indicators in E should be aligned with the corresponding t chunks in C in the same order, which are called alignment templates. Under this scheme, there would be $\|A\| = C_t^d$ number of possible alignments between each pair at template level with respect to different positions of ε.

This method can gurantee each chunk contains two sound units at most. Thus, in a pair of aligned templates, only three mapping forms between sound units are possible: (left is English side)

1. e-to-c1c2: The alignment would be e-to-c1c2 where c1c2 is considered as one unit (initial-final cluster);
2. e1e2-to-c1c2: The alignement would be e1-to-c1 and e2-to-c2.

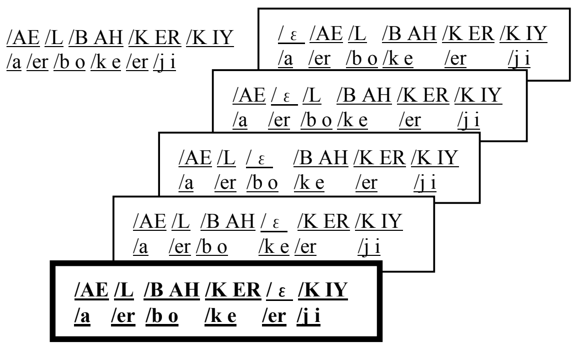
/AE /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i
/ ε /AE /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i
/AE / ε /L /B AH /K ER /K IY
/a /er /b o /k e /er /j i
/AE /L / ε /B AH /K ER /K IY
/a /er /b o /k e /er /j i
/AE /L /B AH / ε /K ER /K IY
/a /er /b o /k e /er /j i
/AE /L /B AH /K ER / ε /K IY
/a /er /b o /k e /er /j i

Figure2. Example of Possible Alignments

3. e1e2-to-c: By adding an additional ε at c side, the alignment would be considered as e1-to-c and e2-to-ε or e1-to-ε and e2-to-c. In this case, we update $||A|| = ||A| + 1$.

For example, the alignment templates between the pair /AE L B AH K ER K IY/ (Albuquerque) and /a er bo ke er ji/ (阿尔伯克尔基) are shown (indicators are tagged using '/') in Figure 2, where 5 possible template alignments can be found. At phoneme leve, because of the existence of /K ER/ - /er/ in the first four template alignments, total possible phoneme alignments would be 9.

4.2 EM Training for Symbol-Mapping Probabilities

We then applied expectation-maximization (EM) algorithm to find the best alignment (Viterbi alignment) for each training pair and generate symbol-mapping probabilities. The training goes as the following 4 steps iteratively:

1). Initialization: For each English-Chinese pair, assign equal weights to all alignments generated from 4.1 as $||A||^{-1}$.

2). Expectation Step: For each of the 40 English phonemes, count up instances of its different mappings from the observations on all alignments produced in 4.1. Each alignment contributes counts in proportion to its own weight. Normalized the scores of the Chinese pinyin units it maps to so that the mapping probability sum to 1.

3). Maximization Step: Recompute the alignment scores. Each alignment is scored with the product of the scores of the symbol mappings it contains. Normalize the alignment scores so that each pair's alignments scores sum to 1.

4). Repeat step 2-3 until the symbol-mapping probabilities converge, meaning that the variation of each probability between two iterations becomes less than a specified threshold.

With 1500 pairs of training instances, our aligning algorithm generates 3103 alignments in total. For each English-Chinese pair, its most preferable alignment is found whose alignment score (weight) approaches to 1 with the increase of iteration time. Table 1 shows sample alignments found automatically through EM training. For each English phoneme, top-4 pinyin symbol-mapping probabilities are shown in Table 2.

English name	Chinese Translation	Best Alignment Found
Abel	亚伯	EY B AH L ya b o ε
Agamemnon	亚格门农	AE G AH M EH M N AA N ya g e men ε n o ng
AIDS	艾滋	EY D Z ai ε zi
Albert	艾伯特	AE L B ER T ai ε b o te
Alexander	亚历山大	AE L AH G Z AE N D ER ya l i ε sh a n d a
Alfred	艾佛烈	AE L F R AH D ai ε fo l ie ε
Algiers	阿尔及尔	AE L JH IH R Z a er j i er ε

Table 1. Viterbi alignments learned by EM Training

Dumb sound ε is introduced to both sets of phonetic alphabets due to the preprocessing. It plays an important role in transliteration for carrying out "virtual mapping" with respect to the mapping conditions of 3 and 4 in Section 4.2.

Our EM training initiated with the alignment templates automatically calculates the probabilities of mappings from English phonemes to not only individual pinyin initials and finals, but also initial-final clusters, which are hard to deal with using other methods. From our training instances, we identify 107 of these clusters, among which about 20 are most commonly seen, e.g. /d-e/, /k-e/, /s-i/, /t-e/, etc. They can be included in pinyin inventory as additional sound units and candidates for transcriptions from English sound like /D/, /K/, /S/, /T/, etc. in certain situations. This can significantly decrease the uncertainty caused by taking the finals in these clusters for zero *fertility* in source-channel model.

4.3 WFST for Phonetic Transition

We build a weighted finite state transducer (WFST) based on symbol-mapping probabilities, for the transcription of an input English phoneme sequence into its possible pinyin symbol sequences. Each arc carries the transition information

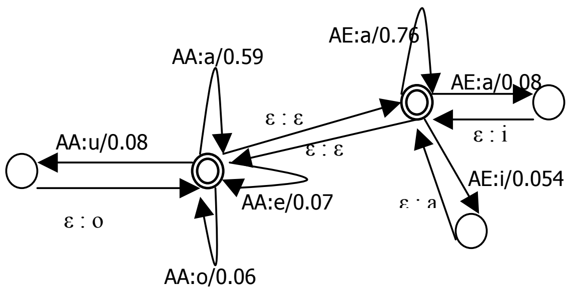

Figure 4. Part of WFST based on P(C|E)

from an English phoneme to its pinyin counterparts as well as their transition probabilities. Figure 4 shows part of the transducer, only including the transitions from two English phonemes /AA/ and /AE/. Note that the arc [/AA/:/uo/|0.085] is split into two arcs, i.e. [/AA/:/u/|0.086] and [/ε/:/o/|1.0] jointed by an intermediate node. This is for the following pinyin syllable segmenter being able to easily connect with it.

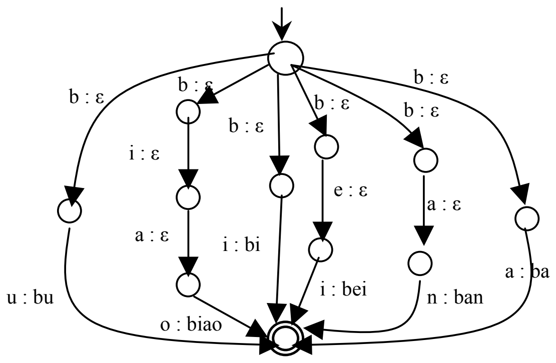

Figure 5. Part of FST for pinyin correction

4.4 Issues of Illegal Pinyin Syllables

Many of pinyin symbol sequences produced by the transliteration model cannot be correctly syllabified or include illegitimate pinyin syllables as the transducer itself has no knowledge about pinyin's regulations. Actually only 396 of 23*25 possible combinations of initials and finials can constitute legal pinyin syllables. Legal syllables, regardless of dialects, can be easily collected from Chinese lexicons in corresponding romanization systems by using an automatic scanning program. Based on this knowledge, we construct a FST as shown in Figure 5, where several syllables beginning with sound /b/ are presented. It is composed with the previous WFST for producing legal pinyin symbol sequences and segmenting them into syllables.

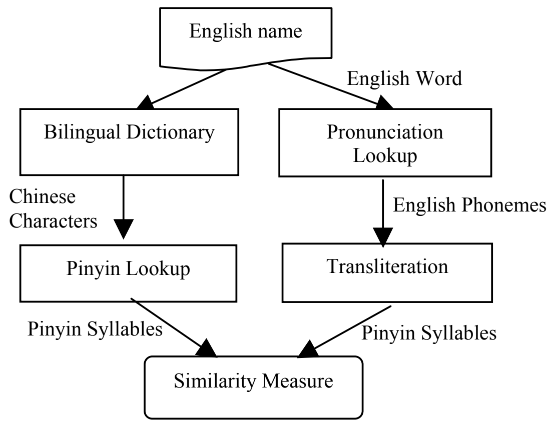

Figure 6. Flowchart of Similarity Test

4.5 Language Model Training

A syllable-based bi-gram language model for pinyin is trained using the same instances of 1500 transliterated names, on which the transliteration WFST is built. The model *P(C)* is approximated by counting the frequencies of syllable occurrence in this data set in terms of the equation:

$$p(C) \cong \prod_i p(c_i \mid c_{i-1}) \cong \prod_i \frac{count(c_i c_{i-1})}{count(c_i)} \qquad \text{(Eq-5)}$$

where c_i is the pinyin syllable of Chinese character.

5 Experiments and Evaluation

5.1 Similarity Measurement

We apply the phoneme-based similarity measurement to transliterations. Figure 6 illustrates the process with the participation of an English-Chinese bilingual dictionary (LDC's bilingual named entity list), an English pronunciation dictionary (CMU's pronunciation dictionary) and a Chinese character-pinyin table (LDC's Chinese character table with pinyin).

Similarity measurement is based on edit distance, which is defined as the minimum number of insertions, deletions and substitutions required for transforming one string to the other. Generally, given two arbitrary strings S1 and S2 from the same character set, a two dimensional matrix, M[0...|S1|, 0...|S2|] where |S1| and |S2| represent the length of the two strings, is used to hold the edit distance scores. The elements in the whole matrix M[,] can be computed row by row by dynamic programming.

The performance of transliteration is measured in terms of error rate. Given S1 is the machine generated and S2 the standard, the error rate of transliteration is defined as

$$e = \frac{d(S_1, S_2)}{|S_2|} \qquad \text{(Eq-6)}$$

where *d(S1, S2)* denotes the edit distance between S1 and S2, and |S2| denotes the length of standard transliteration.

5.2 Experimental Results

5.2.1 Experiment I

In this test, we choose 200 English names from LDC's bilingual named entity list, in which 100 of them are seen in the training set (for close test) and 100 of them are untrained (for open test). Only top-1 machine generated transliteration of each name is chosen for comparison with its standard translation. The error rate distribution is divided into 4 segments ranging in 0%~10%, 10%~30%, 30%~50% and 50%~100%. We count the number of names whose transliteration error rate falls in each segment in the first two rows in Table 3 and present their average percentage in the third row.

Error rate (%)	0~10	10~30	30~50	>50
Untrained (100)	7	24	11	58
Trained (100)	10	21	35	34
Average (200)	8.5%	22.5%	23%	46%

Table 3. Result of Experiment I

5.2.2 Experiment II

We use the same 200 names in Experiment I for carrying out this test. This time, we ask the system to produce up to top-20 (5, 10, 15 and 20) candidates for each input name. We count for any transliteration whose error rate is less than 10% (i.e. 1 out of 10 characters is mismatched in comparison with the standard transliteration), and we consider such transliterations correct. The number of correct transliterations is counted and listed in the first two rows as in Table 4 and their average percentage over 200 names is presented as in the third row. This test evaluates the proportion of instances whose correct transliteration can be found in top-n generated candidates.

Top n	5	10	15	20
Untrained (100)	14	19	23	29
Trained (100)	29	35	51	64
Overall (200)	21.5%	27%	37%	46.5%

Table 4. Result of Experiment II

5.2.3 Experiment III

For comparison with other comparable systems, we carry out a test based on similar setup described in (Meng et al., 2001) and (Virga and Khudanpur, 2003). Their experiments for evaluating transliteration accuracy applied the same data set. We don't have that data set. So we use another data set with the same size instead, i.e. 2233 instances for training and 1541 for testing. We choose the top-1 machine-generated transliteration of each test instance. The average error rate over all testing instances is shown in Table 5.

Systems	Meng et al	Virga et al	Ours
Error rate	52.5%	50.8%	54.2%

Table 5. Transliteration error rate compared to other systems with the same size of data set

Note that the transliteration model and language model are both trained on the same set of 2233 instances. In order to test the influence of the transliteration model and language model separately, we first replace the transliteration model with the one in Experiment I and II, which is trained from 1500 transliterated names, for another test. The error rate becomes 56.5%. And then, we replace the language model with that of Experiment I and II. This time, the error rate of our system rises up to 67.7%.

5.3 Discussions

From the observations on the result of Experiment I where only top-1 candidate is chosen, we note that 54% (100% - 46% = 54%) of the test instances can be transformed to their Chinese translations with an error rate less than 50%. This indicates that majority of the machine generated-transliterations tends to be "half-correct" at least. The close test, where 66% (100-34) machine-generated transliterations with error rate less than 50%, performs better than open test where 42% (100-58) is with the error rate < 50%. Moreover, the possibility of finding the correct transliteration in top-1 result candidates is fairly low on both tests, which can be seen from the indications that only 7% and 10% of the test instances (8.5% in average) end up with the top-1 transliteration that can be considered as correct (error rate < 10%).

From the result of Experiment II where top-1~top-20 transliterations are investigated, we observe that 46.5% (average of untrained and trained instances) of testing instances can have their correct Chinese transliterations (error rate < 10%) within top-20 transliteration candidates. We note that performance on trained test instances is obviously better than that of untrained instances. This result from a lot of similarly pronounced foreign names exist in our training set. Thus, for the trained instances, our language model doesn't suffer from data sparseness.

From Experiment III, we find that the accuracy of our system is comparable with other state-of-the-art systems, but slightly worse. The error rates of three systems show that the top-1 transliterations is almost incorrect since two pinyin sequences with 50% difference in edit distance the (error rate = 50%) represent nearly different names phonetically like /ya li shan da/ (亚力山大) and /ya li shi duo de/ (亚里士多德). Furthermore, the error rate of our system rises from 54.2% to 56.5% as the transliteration model is trained on the data for Experiment I and II, whereas rising to 66.7% when the language model is replaced instead. This indicates that our approach is more sensitive to language model than transliteration model. The reason is that transliteration model only reflects symbol-mapping relationships among phonemes rather than sequences, leaving lots of work done by language model. The problem is supposed to be alleviated by improving transliteration model further.

The presented low accuracy may result from several reasons:

First, our transliteration model does not take the context information into consideration. The neighboring elements of pronunciation units being transcribed can provide significant information to determining their mapping probabilities. For example, both /AH P AA L OW/ (Apollo), which is usually translated to /a bo luo/ (阿波罗), and /EY D AH N/ (Aden), which is translated to /ya ding/ (亚丁), contain the sound /AH/. But in different neighboring context, they are mapped to different pinyin sound, i.e. /a/ and /i(ng)/ respectively.

Second, the data sparseness can significantly affect the search for correct output sequences by using language model. For example, if there were not enough instances covering the syllables /ya li .../ (亚力…) in the training set, it would be hard for the candidate of /ya li shan da/ (亚力山大) to outreach other candidates on the product of their bigram probabilities.

6 Conclusion

We model the statistical transliteration problem as a direct phonetic symbol transcription model plus a language model for post-adjustment. The baseline indicates a comparable performance suggested by other systems. The advantage of direct method is its flexibility for incorporating features with respect to dependencies among surrounding phonemes, for which we can expand our transliteration model in the future by considering context information in terms of feature functions within Maximum Entropy framework (Berger et al, 1996). Also, we will improve our language model using tri-gram for a better accuracy and smoothing techniques for overcoming data sparseness.

Acknowledgements

The work described in this paper is partially supported by CUHK under the Strategic Grant initiative (project account no.: 4410001). We are especially grateful to Prof. Helen Meng for her support and help on obtaining LDC corpora.

e	*c*	P(*c*|*e*)	*c*	P(*c*|*e*)	*c*	P(*c*|*e*)	*c*	P(*c*|*e*)
AA	a	0.599	uo	0.085	e	0.072	o	0.061
AE	a	0.765	ai	0.086	ia	0.054	ya	0.049
AH	a	0.344	i	0.137	u	0.119	e	0.099
AO	uo	0.246	o	0.203	ao	0.174	e	0.159
AW	u	0.643	a	0.214	ei	0.071	uo	0.071
AY	ai	0.612	i	0.163	a	0.069	ei	0.041
B	b	0.737	bu	0.178	bo	0.016	p	0.016
CH	q	0.355	ch	0.290	j	0.129	sh	0.065
D	d	0.623	de	0.202	ε	0.116	t	0.022
DH	si	1.000						
EH	ai	0.312	ei	0.174	e	0.123	a	0.065
ER	e	0.263	o	0.225	a	0.116	u	0.093
EY	ai	0.321	a	0.256	ei	0.231	i	0.077
F	f	0.625	fu	0.264	fo	0.069	fa	0.028
G	g	0.407	ge	0.241	j	0.1667	ε	0.111
HH	h	0.810	x	0.114	xiu	0.038	ε	0.025
IH	i	0.581	ei	0.156	yi	0.138	ε	0.031
IY	i	0.696	ei	0.094	ε	0.060	yi	0.045
JH	j	0.243	y	0.186	zh	0.171	q	0.114
K	k	0.468	ke	0.193	j	0.154	g	0.063
L	l	0.553	er	0.354	ε	0.054	li	0.016
M	m	0.590	mu	0.137	n	0.137	ng	0.085
N	n	0.655	ng	0.160	ε	0.125	en	0.027
NG	n	0.550	ng	0.450				
OW	uo	0.287	e	0.278	o	0.148	ε	0.055
OY	uo	0.429	o	0.286	ei	0.143	v	0.142
P	b	0.324	p	0.297	pu	0.270	ε	0.054
R	l	0.565	ε	0.312	er	0.082	r	0.016
S	si	0.523	s	0.166	x	0.080	ε	0.057
SH	sh	0.294	shi	0.176	x	0.176	sai	0.059
T	d	0.335	t	0.254	te	0.243	ε	0.115
TH	s	0.393	si	0.250	ε	0.077	d	0.071
UH	u	0.882	ou	0.059	ε	0.059		
UW	u	0.609	ε	0.132	ou	0.056	o	0.037
V	w	0.759	f	0.148	fu	0.074	wei	0.019
W	w	0.586	a	0.138	h	0.137	ε	0.034
Y	y	0.500	ε	0.213	k	0.125	n	0.083
Z	si	0.483	s	0.115	ε	0.102	zi	0.092
ZH	x	0.750	r	0.250				
ε	n	0.257	yi	0.121	er	0.106	a	0.099

Table 2. English Phonemes with probabilistic mappings to Chinese pinyin sound units

References

A. L. Berger, S.A. Della Pietra and V.J. Della Pietra, 1996, "A maximum entropy approach to natural language processing." *Computational Linguistics*, 22(1):39-72, March.

P.F. Brown, S.A. Della Pietra, V.J. Della Pietra and R.L. Mercer, 1993, "The mathematics of statistical machine translation: Parameter estimation." *Computational Lingustics*, 19(2):263-311.

K. Knight and J. Graehl, 1997, "Machine transliteration", *Proceedings of the Conference of the Association for Computation Linguistics (ACL).*

W.H. Lin and H.H. Chen, 2002, "Backward machine transliteration by learning phonetic similarity", *Proceedings of CoNLL-2002*, Taipei, Taiwan, pp. 139-145.

H.M. Meng, W.K. Lo, B. Chen and K. Tang, 2001, "Generating phonetic cognates to handle named entities in English-Chinese cross-language spoken document retrieval", *Proceedings of the Automatic Speech Recognition and Understanding Workshop.*

F.J. Och, H. Ney, 2002, "Discriminative training and maximum entropy models for statistical machine translation", In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics*, pp:295-302, Philadelphia, PA, July.

K.A. Papineni, S. Roukos and R.T. Ward, 1998, "Maximum likelihood and discriminative training of direct translation models." In *Proc. Int. Conf. On Acoustics, Speech, and Singal Processing*, pp:189-192, Seattle, WA, May.

P. Virga and S. Khudanpur, 2003, "Transliteration of proper names in cross-lingual information retrieval." *Proceedings of the ACL Workshop on Multi-lingual Named Entity Recognition 2003.*

S. Wan and C.M. Verspoor, 1998, "Automatic English-Chinese name transliteration for development of multilingual resources", *the Joint Meeting of 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics.*