Learning to Find English to Chinese Transliterations on the Web

Jian-Cheng Wu
Department of Computer Science
National Tsing Hua University
101, Kuangfu Road, Hsinchu, Taiwan
d928322@oz.nthu.edu.tw

Jason S. Chang
Department of Computer Science
National Tsing Hua University
101, Kuangfu Road, Hsinchu, Taiwan
jschang@cs.nthu.edu.tw

Abstract

We present a method for learning to find English to Chinese transliterations on the Web. In our approach, proper nouns are expanded into new queries aimed at maximizing the probability of retrieving transliterations from existing search engines. The method involves learning the sublexical relationships between names and their transliterations. At run-time, a given name is automatically extended into queries with relevant morphemes, and transliterations in the returned search snippets are extracted and ranked. We present a new system, *TermMine*, that applies the method to find transliterations of a given name. Evaluation on a list of 500 proper names shows that the method achieves high precision and recall, and outperforms commercial machine translation systems.

1 Introduction

Increasingly, short passages or web pages are being translated by desktop machine translation software or are submitted to machine translation services on the Web every day. These texts usually contain some proportion of proper names (e.g., place and people names in "The cities of Mesopotamia prospered under Parthian and Sassanian rule."), which may not be handled properly by a machine translation system. Online machine translation services such as *Google Translate*[1] or *Yahoo! Babelfish*[2] typically use a bilingual dictionary that is either manually compiled or learned from a parallel corpus. However, such dictionaries often have insufficient coverage of proper names and technical terms, leading to poor translation performance due to out of vocabulary (OOV) problem.

Handling name transliteration is also important for cross language information retrieval (CLIR) and terminology translation (Quah 2006). There are also services on the Web specifically targeting transliteration aimed at improving CLIR, including *CHINET* (Kwok et al. 2005) and *LiveTrans* (Lu, Chien, and Lee 2004).

The OOV problems of machine translation (MT) or CLIR can be handled more effectively by learning to find transliteration on the Web. Consider the sentence in Example (1), containing three proper names. *Google Translate* produces the sentence in Example (2) and leaves "Parthian" and "Sassanian" not translated. A good response might be a translation like Example (3) with appropriate transliterations (underlined).

(1) *The cities of Mesopotamia prospered under Parthian and Sassanian rule.*

(2) 城市繁榮下 parthian 達米亞、sassanian 統治。

(3) <u>美索不達米亞</u>[3]城市在<u>巴底亞</u>[4]和<u>薩珊</u>[5]統治下繁榮起來。

These transliterations can be more effectively retrieved from mixed-code Web pages by extending each of the proper names into a query. Intuitively, by requiring one of likely transliteration morphemes (e.g., "巴"(Ba) or "帕"(Pa) for names beginning with the prefix "par-"), we can bias the search engine towards retrieving the correct trans-

[1] Google Translate: translate.google.com/translate_t

[2] Yahoo! Babelfish: babelfish.yahoo.com

[3] 美索不達米亞(Meisuobudamiya) is the transliteration of "Mesopotamia."

[4] 巴底亞(Badiya) is the transliteration of "Parthian."

[5] 薩珊(Sashan) is the transliteration of "Sassanian."

Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 996–1004, Prague, June 2007.

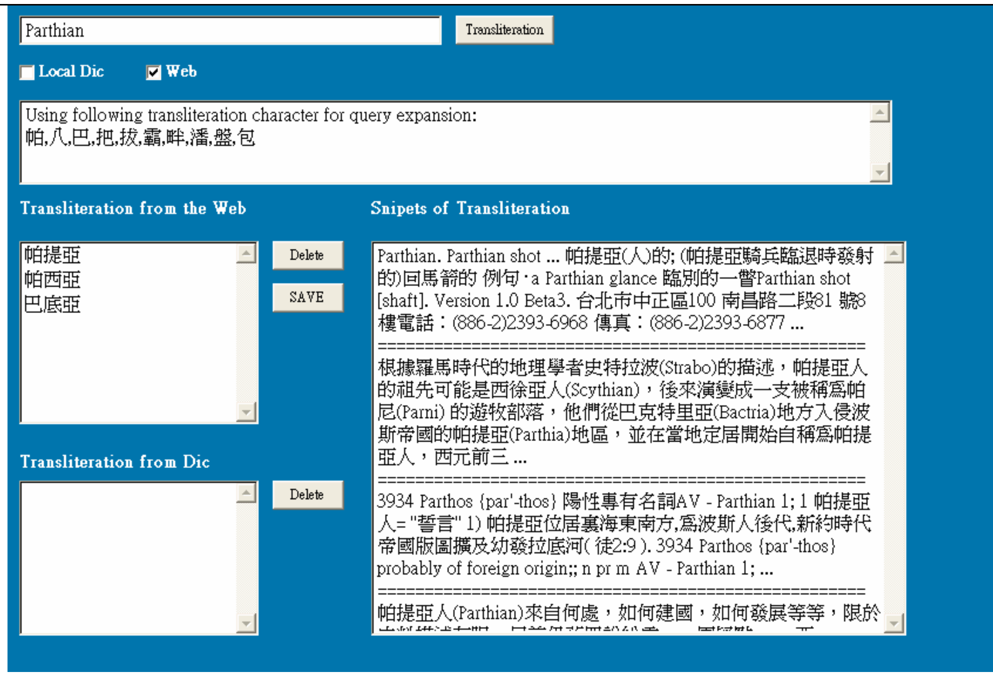

Figure 1. An example of *TermMine* search for transliterations of the name "Parthian"

literations (e.g., "巴底亞"(Badiya) and "帕提亞"(Patiya)) in snippets of many top-ranked documents.

This approach to terminology translation by searching is a strategy increasingly adopted by human translators. Quah (2006) described a modern day translator would search for the translation of a difficult technical term such as "異方性導電樹脂フィルム" by expanding the query with the word "*film*" (back transliteration of the component "フィルム" of the term in question). This kind of query expansion (QE) indeed increases the chance of finding the correct translation "*anisotropic conductive film*" in top-ranked snippets. However, the manual process of expanding query, sending search request, and extracting transliteration is tedious and time consuming. Furthermore, unless the query expansion is done properly, snippets containing answers might not be ranked high enough for this strategy to be the most effective.

We present a new system, *TermMine,* that automatically learns to extend a given name into a query expected to retrieve and extract transliterations of the proper name. An example of machine transliteration of "Parthian" is shown in Figure 1. *TermMine* has determined the best 10 query expansions (e.g., "Parthian 巴," "Parthian 帕"). *TermMine* learns these effective expansions automatically during training by analyzing a collection of place names and their transliterations, and deriving cross-language relationships of prefix and postfix morphemes. For instance, *TermMine* learns that a name that begins with the prefix "par-" is likely to have a transliteration beginning with "巴" or "帕"). We describe the learning process in Section 3.

This prototype demonstrates a novel method for learning to find transliterations of proper nouns on the Web based on query expansion aimed at maximizing the probability of retrieving transliterations from existing search engines. Since the method involves learning the morphological relationships between names and their transliterations, we refer to this IR-based approach as *morphological query expansion approach to machine transliteration*. This novel approach is general in scope and can also be applied to *back transliteration* and to *translation* with slight modifications, even though we focus on transliteration in this paper.

The remainder of the paper is organized as follows. First, we give a formal statement for the problem (Section 2). Then, we present a solution to the problem by proposing new transliteration probability functions, describing the procedure for estimating parameters for these functions (Section 3) and the run-time procedure for searching and extracting transliteration via a search engine (Section 4). As part of our evaluation, we carry out two sets of experiments, with or without query expansion, and compare the results. We also evaluate the results against two commercial machine translation online services (Section 5).

2 Problem Statement

Using online machine translation services for name transliteration does not work very well. Searching in the vicinity of the name in mixed-code Web pages is a good strategy. However, query expansion is needed for this strategy to be effective. Therefore, to find transliterations of a name, a promising approach is to automatically expand the given name into a query with the additional requirement of some morpheme expected to be part of relevant transliterations that might appear on the Web.

Table 1. Sample name-transliteration pairs from the training collection.

Name	Transliteration	Name	Transliteration
Aabenraa	阿本洛	Aarberg	阿爾柏
Aabybro	阿比布洛	Aarburg	亞爾堡
Aachen	亞琛	Aardenburg	亞丁堡
Aalesund	奧勒孫	Aargau	亞高
Aaley	阿利	Aars	阿爾斯
Aalten	阿爾廷	Aba	阿巴
Aarau	亞牢	Abacaxis	阿巴卡克斯

Now, we formally state the problem we are dealing with:

> While a *proper name N* is given. Our goal is to search and extract the *transliteration T* of *N* from Web pages via a general-purpose *search engine SE*. For that, we expand *N* into a set of queries q_1, q_2, …, q_m, such that the top *n* document snippets returned by *SE* for the queries are likely to contain some transliterations *T* of the given name *N*.

In the next section, we propose using a probability function to model the relationships between *names* and *transliterations* and describe how the parameters in this function can be estimated.

3 Learning Relationships for QE

We attempt to derive cross-language morphological relationships between names and transliterations and use them to expand a name into an effective query for searching and extracting transliterations. For the purpose of expanding the given name, *N,* into effective queries to search and extract transliterations *T*, we define a probabilistic function for mapping prefix syllable from the source to the target languages. The prefix transliteration function $P(T_P \mid N_P)$ is the probability of *T* has a prefix T_P under the condition that the name *N* has a prefix N_P.

$$P(T_P \mid N_P) = \text{Count}(T_P, N_P) / \text{Count}(N_P) \qquad (1)$$

where Count (T_P,N_P) is the number of T_P and N_P co-occurring in the pairs of training set (see Table 1), and Count(N_P) is the number of N_P occurring in training set.

Similarly, we define the function $P(T_S \mid N_S)$ for postfixes T_S and N_S:

$$P(T_S \mid N_S) = \text{Count}(T_S, N_S) / \text{Count}(N_S) \qquad (2)$$

The prefixes and postfixes are intended as a syllable in the two languages involved, so the two prefixes correspond to each other (See Table 2&3). Due to the differences in the sound inventory, the Roman prefix corresponding to a syllabic prefix in Chinese may vary, ranging from a consonant, a vowel, or a consonant followed by a vowel (but not a vowel followed by a consonant). So, it is likely such a Roman prefix has from one to four letters. On the contrary, the prefix syllable for a name written in Chinese is readily identifiable.

Table 2. Sample cross-language morphological relationships between prefixes.

Name Prefix (N_P)	Transliteration Prefix (T_P)	N_P Count	T_P Count	Co-occ. Count
a-	阿(A)	1,456	854	854
a-	亞(Ya)	1,456	267	264
ab-	阿(A)	77	854	45
ab-	亞(Ya)	77	267	32
b-	布(Bu)	2,319	574	566
b-	巴(Ba)	2,319	539	521
ba-	巴(Ba)	650	574	452
bu-	布(Bu)	299	539	182

Table 3. Sample cross-language morphological relationships between postfixes.

Name Postfix (N_s)	Transliteration Postfix (T_s)	N_s Count	T_s Count	Co-occ. Count
-a	拉(La)	4,774	1,044	941
-a	亞(Ya)	4,774	606	568
-la	拉(La)	461	1,044	422
-ra	拉(La)	534	1,044	516
-ia	亞(Ya)	456	606	391
-nia	亞(Ya)	81	606	77
-burg	堡(Bao)	183	230	175

We also observe that a preferred prefix (e.g., “艾”(Ai)) is often used for a Roman prefix (e.g., “a-” or “ir-”), while occasionally other homophonic characters are used (e.g., “埃”(Ai)). The skew distribution creates problems for reliable estimation of transliteration functions. To cope with this data sparseness problem, we use homophone classes and a function CL that maps homophonic characters to the same class number. For instance, “艾” and “埃” are homophonic, and both are assigned the same class identifier(see Table 4 for more samples).

Therefore, we have

CL (“艾”) = CL (“埃”) = 275.

Table 4. Some examples of classes of homophonic characters. The class ID of each class is assigned arbitrarily.

Class ID	Transl. char	Pronunciation	Class ID	Transl. char	Pronunciation
1	八	Ba	2	波	Bo
1	巴	Ba	275	艾	Ai
1	拔	Ba	275	埃	Ai
1	把	Ba	275	愛	Ai
1	罷	Ba	276	敖	Ao
1	霸	Ba	276	奧	Ao
2	白	Bo	276	澳	Ao
2	伯	Bo	…	…	…

With homophonic classes of transliteration morphemes, we define class-based transliteration probability as follows

$$P_{CL}(C \mid N_P) = \text{Count}(T_P, N_P) \,/\, \text{Count}(N_P) \qquad (3)$$

where $CL(T_P) = C$

$$P_{CL}(C \mid N_S) = \text{Count}(T_S, N_S) \,/\, \text{Count}(N_S) \qquad (4)$$

where $CL(T_S) = C$

and then we rewrite $P(T_P \mid N_P)$ and $P(T_S \mid N_S)$ as

$$P(T_P \mid N_P) = P_{CL}(CL(T_P) \mid N_P) \qquad (5)$$

$$P(T_S \mid N_S) = P_{CL}(CL(T_S) \mid N_S) \qquad (6)$$

With class-based transliteration probabilities, we are able to cope with difficulty in estimating parameters for rare events which are under represented in the training set. Table 5 shows that “埃” belongs to a homophonic class co-occurring with “a-” for 46 times, even when only one instance of (“埃”, “a-”).

After cross-language relationships for prefixes and postfixes are automatically trained, the prefix relationships are stored as prioritized query expansion rules. In addition to that, we also need a transliteration probability function to rank candidate transliterations at run-time (Section 4). To cope with data sparseness, we consider names (or transliterations) with the same prefix (or postfix) as a class. With that in mind, we use both prefix and postfix to formulate an interpolation-based estimator for name transliteration probability:

$$P(T \mid N) = \max_{N_P, N_S} \lambda_1 P(T_P \mid N_P) + \lambda_2 P(T_S \mid N_S) \qquad (7)$$

where $\lambda_1 + \lambda_2 = 1$ and N_P, N_S, T_P, and T_S are the prefix and postfix of the given name N and transliteration T.

For instance, the probability of “美索不達米亞”(Meisuobudamiya) as a transliteration of “*Mesopotamia*” is estimated as follows

$$P(\text{美索不達米亞} \mid \text{“}Mesopotamia\text{”}) = \lambda_1 P(\text{“美”} \mid \text{“me-”}) + \lambda_2 P(\text{“亞”} \mid \text{“-a”})$$

(1) For each entry in the bilingual name list, pair up prefixes and postfixes in names and transliterations.
(2) Calculate counts of these affixes and their co-occurrences.
(3) Estimate the prefix and postfix transliteration functions
(4) Estimate class-based prefix and postfix transliteration functions

Figure 2. Outline of the process used to train the *TermMine* system.

The system follows the procedure shown in Figure 2 to estimate these probabilities. In Step (1), the system generates all possible prefix pairs for each name-transliteration pair. For instance, consider the pair, (“Aabenraa,” “阿本洛”), the system will generate eight pairs:

(a-, 阿-), (aa-, 阿-), (aab-, 阿-), (aabe-, 阿-),
(-a, -洛), (-aa, -洛), (-raa, -洛), and (-nraa, -洛).

Finally, the transliteration probabilities are estimated based on the counts of prefixes, postfixes, and their co-occurrences. The derived probabilities embody a number of relationships:

(a) Phoneme to syllable relationships (e.g., “b” vs. “布” as in “Brooklyn” and “布魯克林”(Bulukelin)),
(b) Syllable to syllable relationships (e.g., “bu” vs. “布”),
(c) Phonics rules (e.g., “br-“ vs. “布” and “克” vs. “cl-”). The high probability of P(“克” | “cl-”) amounts to the phonics rule that stipulates “c” be pronounced with a “k” sound in the context of “l.”

4 Transliteration Search and Extraction

At run-time, the system follows the procedure in Figure 3 to process the given name. In Step (1), the system looks up in the prefix relationship table to find the n best relationships (n = *MaxExpQueries*) for query expansion with preference for relationships with higher probabilistic value. For instance, to search for transliterations of “Acton,” the system looks at all possible prefixes and postfixes of “Acton,” including *a-*, *ac-*, *act-*, *acto-*, *-n*, *-on*, *-ton*, and *-cton*, and determines the best query expansions: “Acton 阿,” “Acton 亞,” “Acton 艾,” “Acton 頓,” “Acton 騰,” etc. These effective expansions are automatically derived during the training stage described in Section 3 by analyzing a large collection of name-transliteration pairs.

In Step (2), the system sends off each of these queries to a search engine to retrieve up to *MaxDocRetrieved* document snippets. In Step (3), the system discards snippets that have too little proportion of target-language text. See Example (4) for a snippet that has high portion of English text and therefore is less likely to contain a transliteration. In Step (4), the system considers the substrings in the remaining snippets.

(1) Look up the table for top *MaxExpQueries* prefix and posfix relationships relevant to the given name and use the target morphemes in the relationship to form expanded queries
(2) Search for Web pages with the queries and filter out snippets containing at less than *MinTargetRate* portion of target language text
(3) Evaluate candidates based on class-based transliteration probability (Equation 5)
(4) Output top one candidate for evaluation

Figure 3. Outline of the steps used to search, extract, and rank transliterations.

Table 5. Sample data for class-based morphological transliteration probability of prefixes, where # of N_P denotes the number of the name prefix N_P; # of C, N_P denotes the number of all T_P belonging to the class C co-occurring with the N_P; # T_P, N_P denotes the number of transliteration prefix T_P co-occurs with the N_P; $P(C|N_P)$ denotes the probability of all T_P belonging to C co-occurring with the N_P; $P(T_P|N_P)$ denotes the probability of the T_p co-occurs with the N_P.

N_P	Class ID	T_P	# of N_P	# of C,N_P	# of T_P,N_P	$P(C\|N_P)$	$P(T_P\|N_P)$
a-	275	艾	1456	46	28	0.032	0.019
a-	275	愛	1456	46	17	0.032	0.012
a-	275	埃	1456	46	1	0.032	0.000
a-	276	奧	1456	103	100	0.071	0.069
a-	276	澳	1456	103	2	0.071	0.001
a-	276	敖	1456	103	1	0.071	0.000
ba-	2	波	652	5	3	0.008	0.005
ba-	2	百	652	5	1	0.008	0.002
ba-	2	柏	652	5	1	0.008	0.002

Table 6. Sample data for class-based morphological transliteration probability of postfixes. Notations are similar to those for Table 5.

N_s	Class ID	T_s	# of N_s	# of C,N_s	# of T_s,N_s	$P(C\|N_s)$	$P(T_s\|N_s)$
-li	103	利	142	140	85	0.986	0.599
-li	103	里	142	140	52	0.986	0.366
-li	103	力	142	140	2	0.986	0.014
-li	103	立	142	140	1	0.986	0.007
-li	103	李	142	140	0	0.986	0.000
-raa	112	洛	4	1	1	0.250	0.250
-raa	112	珞	4	1	0	0.250	0.000
-raa	112	絡	4	1	0	0.250	0.000
-raa	112	落	4	1	0	0.250	0.000

For instance, Examples (5-7) shows remaining snippets that have high proportion of Chinese text. The strings "阿克頓"(Akedun) is a transliteration found in snippet shown in Example (5), a candidate beginning with the prefix "阿" and ending with the postfix "頓" and is within the distance of 1 of the instance "Acton," separated by a punctuation token. The string "埃克頓" (Aikedun) found in Example (6) is also a legitimate transliteration beginning with a different prefix "埃," while "艾科騰"(Aiketeng) in Example (7) is a transliteration beginning with yet another prefix "艾." Transliteration "埃克頓" appears at a distance of 3 from "Acton," while two instances of "艾科騰" appear at the distances of 1 and 20 from the nearest instances of "Acton."

(4) Acton moive feel pics!! - 攝影
目前位置: 文藝線 > 遊藝支線 > 攝影 > Acton moive feel pics!! Hop Hero - Acton moive feel pics!!
http://www.hkmassive.com/forum/viewthread.php?tid=2368&fpage=1 Watch the slide show! ...

(5) New Home Alert - Sing Tao New Homes
Please select, Acton 阿克頓, Ajax 亞積士, Alliston 阿里斯頓, Ancaster 安卡斯特, Arthur 阿瑟, Aurora 奧羅拉, Ayr 艾爾, Barrie 巴里, Beamsville, Belleville ...

(6) STS-51-F – Wikipedia
前排左起：英格蘭、海因茲、福勒頓、布里奇斯 ... 卡爾·海因茲 (Karl Henize，曾執行 STS-51-F任務)，任務專家; 羅倫·埃克頓 (Loren Acton，曾執行 STS-51-F 任務)，有效載荷專家; 約翰-大衛·巴托 (John-David F. ...

(7) 澳洲艾科騰-00-Acton-Australia.htm
Acton Systems is a world leading manufacturer supplying stuctured cabling systems suited to the Australian and New Zealand marketplace. 澳洲艾科騰乃專業之整合式配線系統製造商, 產品銷售於澳洲及紐西蘭。 Custom made leads are now available ...

The occurrence counts and average distance from instances of the given name are tallied for each of these candidates. Candidates with a low occurrence count and long average distance are excluded from further consideration. Finally, all candidates are evaluated and ranked using Equation (7) given in Section 3.

5 Evaluation

In the experiment carried out to assess the feasibility to the proposed method, a data set of 23,615 names and transliterations was used. This set of place name data is available from NICT, Taiwan for training and testing. There are 967 distinct Chinese characters presented in the data, and more details of training data are available in Table 7. The English part consists of Romanized versions of names originated from many languages, including Western and Asian languages. Most of the time, the names come with a Chinese counterpart based solely on transliteration. But occasionally, the Chinese counterpart is part translation and part transliteration. For instance, the city of "Southampton" has a Chinese counterpart consisting of "南" (translation of "south") and "漢普頓" (transliteration of "ampton").

Table 7. Training data and statistics

Type of Data Used in Experiment	Number
Name-transliteration pairs	23,615
Training data	23,115
Test data	500
Distinct transliteration morphemes	967
Distinct transliteration morphemes (80% coverage)	100
Names with part translation and part transliteration (estimated)	300
Cross-language prefix relationships	21,016
Cross-language postfix relationships	26,564

We used the set of parameters shown in Table 8 to train and run System *TermMine*. A set of 500 randomly selected were set aside for testing. We paired up the prefixes and postfixes in the remaining 23,116 pairs, by taking one to four leading or trailing letters of each Romanized place names and the first and last Chinese transliteration character to estimate P ($T_P | N_P$) and P ($T_S | N_S$).

Table 8. Parameters for training and testing

Parameter	Value	Description
MaxPrefixLetters	4	Max number of letters in a prefix
MaxPostfixLetters	4	Max number of letters in a postfix
MaxExpQueries	10	Max number of expanded queries
MaxDocRetrieved	1000	Max number of document retrieved

MinTargetRate	0.5	Min rate of target text in a snippet
MinOccCount	1	Min number of co-occurrence of query and transliteration candidate in snippets
MaxAvgDistance	4	Max distance between *N* and *T*
WeightPrefixProb	0.5	Weight of Prefix probability (λ_1)
WeightPostfixProb	0.5	Weight of Postfix probability (λ_2)

We carried out two kinds of evaluation on System *TermMine*, with and without query expansion. With QE option off, the name itself was sent off as a query to the search engine, while with QE option turned on, up to 10 expanded queries were sent for each name. We also evaluated the system against *Google Translate* and *Yahoo! Babelfish*. We discarded the results when the names are returned untranslated. After that, we checked the correctness of all remaining results by hand. Table 9 shows a sample of the results produced by the three systems.

In Table 10, we show performance differences of system *TermMine* in query expansion option. Without QE, the system returns transliterations (applicability) less than 50% of the time. Nevertheless, there are enough snippets for extracting and ranking of transliterations. The precision rate of the top-ranking transliterations is 88%. With QE turned on, the applicability rate increases significantly to 60%. The precision rate also improved slightly to 0.89.

The performance evaluation of three systems is shown in Table 11. For the test set of 500 place names, *Google Translate* returned 146 transliterations and *Yahoo! Babelfish* returned only 44, while *TermMine* returned 300. Of the returned transliterations, *Google Translate* and *Yahoo! Babelfish* achieved a precision rate around 50%, while *TermMine* achieved a precision rate almost as high as 90%. The results show that System *TermMine* outperforms both commercial MT systems by a wide margin, in the area of machine transliteration of proper names.

Table 9. Sample output by three systems evaluated. The stared transliterations are incorrect.

Name	***TermMine***	***Google Translate***	***Yahoo! Babelfish***
Arlington	雅靈頓	阿靈頓	阿靈頓
Toledo	托雷多	托萊多	-
Palmerston	帕默斯頓	帕麥斯頓	-
Cootamundra	庫塔曼德拉	庫塔曼德拉	-
Bangui	班基	班吉	-
Australasia	澳大拉西亞	*大洋洲	澳大利西亞
Wilson	威爾森	威爾遜	威爾遜
Mao	*馬寅卯	毛	毛
Inverness	因弗內斯	*禮士	因弗內斯
Cyprus	賽普勒斯	賽普勒斯	塞浦路斯
Rostock	羅斯托克	羅斯托克	羅斯托克
Bethel	貝瑟爾	貝瑟爾	*聖地
Arcade	阿凱德	*商場	*拱廊
Lomonosov	羅蒙諾索夫	羅蒙諾索夫	-
Oskaloosa	奧斯卡盧薩	奧斯卡羅薩	-

Table 10. Performance evaluation of TermMine

Evaluation \ Method	TermMine QE-	TermMine QE+
# of cases performed	238	300
Applicability	0.48	0.60
# Correct Answers	209	263
Precision	0.88	0.89
Recall	0.42	0.53
F-measure	0.57	0.66

Table 11. Performance evaluation of three systems

Evaluation \ Method	**TermMine QE+**	**Google Translate**	**Yahoo! Babelfish**
# of cases done	300	146	44
# of correct answers	263	67	23
Applicability	0.60	0.29	0.09
Precision	0.89	0.46	0.52
Recall	0.53	0.13	0.05
F-measure	0.66	0.21	0.08

6 Comparison with Previous Work

Machine transliteration has been an area of active research. Most of the machine transliteration method attempts to model the transliteration process of mapping between graphemes and phonemes. Knight and Graehl (1998) proposed a multilayer model and a generate-and-test approach to perform back transliteration from Japanese to English based on the model. In our work we address an issue of producing transliteration by way of search.

Goto et al. (2003), and Li et al. (2004) proposed a grapheme-based transliteration model. Hybrid transliteration models were described by Al-Onaizan and Knight (2002), and Oh et al. (2005).

Recently, some of the machine transliteration study has begun to consider the problem of extracting names and their transliterations from parallel corpora (Qu and Grefenstette 2004, Lin, Wu and Chang 2004; Lee and Chang 2003, Li and Grefenstette 2005).

Cao and Li (2002) described a new method for base noun phrase translation by using Web data. Kwok, et al. (2001) described a system called *CHINET* for cross language name search. Nagata et al. (2001) described how to exploit proximity and redundancy to extract translation for a given term. Lu, Chien, and Lee (2002) describe a method for name translation based on mining of anchor texts. More recently, Zhang, Huang, and Vogel (2005) proposed to use occurring words to expand queries for searching and extracting transliterations. Oh and Isahara (2006) use phonetic-similarity to recognize transliteration pairs on the Web.

In contrast to previous work, we propose a simple method for extracting transliterations based on a statistical model trained automatically on a bilingual name list via unsupervised learning. We also carried out experiments and evaluation of training and applying the proposed model to extract transliterations by using web as corpus.

7 Conclusion and Future Work

Morphological query expansion represents an innovative way to capture cross-language relations in name transliteration. The method is independent of the bilingual lexicon content making it easy to adopt to other proper names such person, product, or organization names. This approach is useful in a number of machine translation subtasks, including name transliteration, back transliteration, named entity translation, and terminology translation.

Many opportunities exist for future research and improvement of the proposed approach. First, the method explored here can be extended as an alterative way to support such MT subtasks as back transliteration (Knight and Graehl 1998) and noun phrase translation (Koehn and Knight 2003). Finally, for more challenging MT tasks, such as handling sentences, the improvement of translation quality probably will also be achieved by combining this IR-based approach and statistical machine translation. For example, a pre-processing unit may replace the proper names in a sentence with transliterations (e.g., mixed code text "*The cities of* 美索不達米亞 *prospered under* 巴底亞 *and* 薩珊 *rule*." before sending it off to MT for final translation.

References

GW Bian, HH Chen. Cross-language information access to multilingual collections on the internet. 2000. *Journal of American Society for Information Science & Technology (JASIST), Special Issue on Digital Libraries*, 51(3), pp.281-296, 2000.

Y. Cao and H. Li. Base Noun Phrase Translation Using Web Data and the EM Algorithm. 2002. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02),* pp.127-133, 2002.

PJ. Cheng, JW. Teng, RC. Chen, JH. Wang, WH. Lu, and LF. Chien. Translating unknown queries with web corpora for cross-language information retrieval. 2004. In *Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval (SIGIR04)*, pp. 146-153, 2004.

I. Goto, N. Kato, N. Uratani, and T. Ehara. Transliteration considering context information based on the maximum entropy method. In *Proceedings of Ninth Machine Translation Summit*, pp.125-132, 2003.

F. Huang, S. Vogel, and A. Waibel. Automatic extraction of named entity translingual equivalence based on multi-feature cost minimization. In *Proceeding of the 41st ACL, Workshop on Multilingual and Mixed-Language Named Entity Recognition*, Sapporo, 2003.

A. Kilgarriff and Grefenstette, G. 2003. Introduction to the Special Issue on the Web as Corpus. *Computational Linguistics 29(3)*, pp. 333-348, 2003.

K. Knight, J. Graehl. Machine Transliteration. 1998. *Computational Linguistics 24(4)*, pp.599-612, 1998.

P. Koehn, K. Knight. 2003. Feature-Rich Statistical Translation of Noun Phrases. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 311-318, 2003.

J. Kupiec. 1993. An Algorithm for Finding Noun Phrase Correspondences in Bilingual Corpora. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 17-22, 1993.

KL Kwok. 2001. NTCIR-2 Chinese, Cross Language Retrieval Experiments Using PIRCS. In *Proceedings of NTCIR Workshop Meeting*, pp.111-118, 2001.

KL Kwok, P Deng, N Dinstl, HL Sun, W Xu, P Peng, and Doyon, J. 2005. CHINET: a Chinese name finder system for document triage. In *Proceedings of 2005*

International Conference on Intelligence Analysis, 2005.

C.J. Lee, and Jason S. Chang. 2003. Acquisition of English-Chinese Transliterated Word Pairs from Parallel-Aligned Texts using a Statistical Machine Transliteration Model, In *Proceedings of HLT-NAACL 2003 Workshop*, pp. 96-103, 2003.

H. Li, M. Zhang, and J. Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pp.159-166, 2004.

Y. Li, G. Grefenstette. 2005. Translating Chinese Romanized name into Chinese idiographic characters via corpus and web validation. In *Proceedings of CORIA 2005*, pp. 323-338, 2005.

T. Lin, J.C. Wu, and J. S. Chang. 2004. Extraction of Name and Transliteration in Monolingual and Parallel Corpora. In *Proceedings of AMTA 2004*, pp.177-186, 2004.

WH. Lu, LF. Chien, and HJ. Lee. 2002. Translation of web queries using anchor text mining. *ACM Transactions on Asian Language Information Processing, 1(2)*:159–172, 2002.

WH Lu, LF Chien, HJ Lee. Anchor text mining for translation of Web queries: A transitive translation approach. *ACM Transactions on Information Systems 22(2)*, pp. 242-269, 2004.

M. Nagata, T. Saito, and K. Suzuki. Using the Web as a bilingual dictionary. 2001. In *Proceedings of 39th. ACL Workshop on Data-Driven Methods in Machine Translation*, pp. 95-102, 2001.

J.-H Oh, and H. Isahara. 2006. Mining the Web for Transliteration Lexicons: Joint-Validation Approach, In *IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 254-261, 2006.

J.-H. Oh and K.-S. Choi. 2005. An ensemble of grapheme and phoneme for machine transliteration. In *Proceedings of IJCNLP05*, pp.450–461, 2005.

Y. Qu, and G. Grefenstette. 2004. Finding Ideographic Representations of Japanese Names Written in Latin Script via Language Identification and Corpus Validation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pp.183-190, 2004.

CK Quah. 2006. Translation and Technology, *Palgrave Textbooks in Translation and Interpretation*, Palgrave MacMillan.

Y Zhang, F Huang, S Vogel. 2005. Mining translations of OOV terms from the web through cross-lingual query expansion. In *Proceedings of the 28th Annual International ACM SIGIR*, pp.669-670, 2005.

Y. Zhang and P. Vines. 2004. Detection and translation of oov terms prior to query time. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.524-525, 2004.