Chapter 4: Machine Transliteration

In this chapter, we first give an overview of machine transliteration and briefly illustrate our approach with an example. A formal description of the proposed transliteration model and a parameter estimation procedure based on the EM algorithm will be presented in the subsequent sections. The estimation of LTP (lexical translation probability) and the extraction of transliterated words from aligned sentences based on the proposed transliteration model will also described in this chapter.

4.1 Overview of the Noisy Channel Model

Proper nouns, especially person names, are typically transliterated into phonetic equivalents, when NEs are translated. Since Chinese and English are disparate languages and no simple rules are available for direct mapping between them based on sounds, one possible solution is to adopt a Chinese romanization system[2] to

[2] Ref. sites: “http://www.romanization.com/index.html” and “http://www.edepot.com/taoroman.html”.

represent the pronunciation of each Chinese character and then find the mapping rules between them. Among the many romanization systems for Chinese, Wade-Giles and Hanyu Pinyin are the most widely used. The Wade-Giles system is commonly adopted in Taiwan today and has traditionally been popular among Western scholars. For this reason, we use the Wade-Giles system to romanize Chinese characters. However, the proposed approach is equally applicable to other romanization systems. In the following discussion, *E* and *F* are assumed to be an English word and a romanized Chinese character sequence, respectively. One can consider machine transliteration as a noisy channel, as illustrated in Figure 4.1.

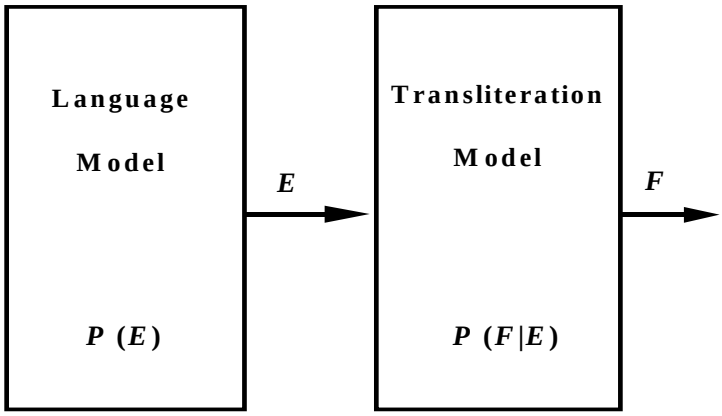

Figure 4.1 The noisy channel model in machine transliteration.

The language model, $P(E)$, generates a source proper name E, and the transliteration model, $P(F|E)$, converts the E into a target transliteration F. $P(E)$ describes the probability associated with E, whereas $P(F|E)$ estimates the probability of F, conditioned on E. $P(F|E)$ can be approximated by decomposing E and F into

transliteration units (TUs). A TU is defined as a sequence of characters transliterated as a group (Lee and Chang, 2003; Lee et al., 2006a). For English, a TU can be a monograph, a digraph, or a trigraph (Wells, 2001). For Chinese, a TU can be a syllable initial, a syllable final, or a syllable (Chao, 1968) represented by romanized characters.

To illustrate how the approach works, take, for example, an English name, "Smith," which can be segmented into four TUs and aligned with the romanized transliteration. Assuming that the word is segmented into "S-m-i-th," then a possible alignment with the Chinese transliteration "史密斯 (Shihmissu)" is depicted in Figure 4.2

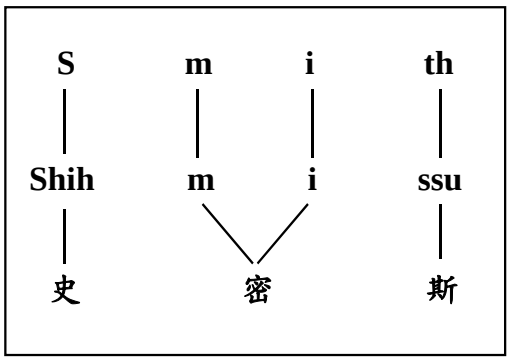

Figure 4.2 TU alignment between English and Chinese romanized character sequences.

Intuitively, the probability of $P(\text{史密斯}|Smith)$ can be simply approximated by Eq. (4.1). A formal description of this approximation scheme will be given in the next subsection:

$$P(\text{史密斯}|\,Smith) \cong P(Shihmissu|\,Smith) \cong P(Shih|\,S)P(m|\,m)P(i|\,i)P(ssu|\,th), \tag{4.1}$$

where “Shihmissu” is the Wade-Giles romanization of “史密斯.”

4.2 Formal Description: Transliteration Model (TM)

A word E with l characters and its romanized word F with m characters are denoted by $E_1\,E_2\,\ldots E_l\,(=E_1^l)$ and $F_1\,F_2\,\ldots F_m\,(=F_1^m)$ respectively. We can represent the mapping of (E, F) as a sequence of matched n TUs, $\{(u_1, v_1), (u_2, v_2), \ldots (u_n, v_n)\}$:

$$\begin{cases} E = E_1E_2\ldots E_l = u_1u_2\ldots u_n \\ F = F_1F_2\ldots F_m = v_1v_2\ldots v_n \end{cases}. \tag{4.2}$$

Hence, the alignment a between E and F can be represented as a match type sequence $(m_1\,m_2\,\ldots m_n)$, where m_i denotes a pair of lengths of u_i and v_i. Therefore, the probability of F given E, $P(F|E)$, is expressed as follows:

$$P(F|\,E) = \sum_a P(F, a|\,E). \tag{4.3}$$

According to the above definitions and independent assumptions, Eq. (4.3) can be further derived as follows:

$$P(F|\,E) = \sum_a P(v_1v_2\ldots v_n|\,u_1u_2\ldots u_n)P(m_1m_2\ldots m_n) = \sum_a \prod_{i=1}^{n} P(v_i|\,u_i)P(m_i). \tag{4.4}$$

Then, to reduce the amount of computation, the process of finding the most probable transliteration F^*, for a given E, can be approximated as

$$F^* = \arg\max_F \max_a P(F, a|\,E) = \arg\max_F \max_a \prod_{i=1}^{n} P(v_i|\,u_i)P(m_i). \tag{4.5}$$

Estimating LTP based on TM

Each of the nouns in the NE phrase being translated may be a common noun or a proper noun. For common nouns, we rely on word alignment to estimate LTP. For proper nouns, we consider machine transliteration for estimation of LTP.

According to Eq. (4.5), the transliteration score function for F, given E, is formulated as

$$Score_{tm}(F \mid E) = \max_{a} \log(\prod_{i=1}^{n} P(v_i \mid u_i) P(m_i)), \qquad (4.6)$$
$$= \max_{a} \sum_{i=1}^{N} (\log P(v_i \mid u_i) + \log P(m_i)).$$

Let $S(i, j)$ be the maximum accumulated log probability between the first i characters of E and the first j characters of F. Then, $\log P(F \mid E) = S(l, n)$, the maximum accumulated log probability among all possible alignment paths of E with length l and of F with length n, can be computed using a dynamic programming strategy, as shown in the following:

Step 1 (Initialization):

$$S(0,0) = 0. \qquad (4.7)$$

Step 2 (Recursion):

$$S(i, j) = \max_{h,k} S(i-h, j-k) + \log P(F_{j-k}^{j} \mid E_{i-h}^{i}) + \log P(h, k), \qquad (4.8)$$
$$0 \le i \le l, \quad 0 \le j \le n.$$

Step 3 (Termination):

$$S(l,n) = \max_{h,k} S(l-h, n-k) + \log P(F_{n-k}^{n} \mid E_{l-h}^{l}) + \log P(h,k), \tag{4.9}$$

where $P(h,k)$ is defined as the probability of the match type "*h-k*."

In practice, the values of *h* and *k* are limited to a small set.

4.3 Estimation of Model Parameters

In the following, we describe the iterative procedure for re-estimation of $P(v_j \mid u_i)$ and $P(m_i)$. We first define the following functions:

$count(u_i, v_j)$ = *the number of occurrences of aligned pair* u_i *and* v_i *in the training set;*

$count(u_i)$ = *the number of occurrences of* u_i *in the training set;*

$count(h,k)$ = *the total number of occurrences of match type "h-k" in the training set.*

Therefore, the probabilities $P(v_j \mid u_i)$ and $P(h,k)$ can be approximated as follows:

$$P(v_j \mid u_i) = \frac{count(u_i, v_j)}{count(u_i)}. \tag{4.10}$$

$$P(h,k) = \frac{count(h,k)}{\sum_i \sum_j count(i,j)}. \tag{4.11}$$

Because $count(u_i, v_j)$ is unknown at the beginning, a reasonable approach to obtaining an initial estimate of the parameters of the translation model is to constrain the TU alignments of a word pair (*E*, *F*) within a position distance δ (Lee and Choi,

1997). Assume that $u_i = E_p^{p+h-1}$ and $v_j = F_q^{q+k-1}$, and that $d_\delta(u_i, v_j)$ is the allowable position distance within δ for the aligned pair (u_i, v_i). $d_\delta(u_i, v_j)$ is defined as follows:

$$d_\delta(u_i, v_j) = \begin{cases} \left| p - \frac{q \times l}{n} \right| < \delta, & and \\ \left| (p+h-1) - \frac{(q+k-1) \times l}{n} \right| < \delta \end{cases}, \quad (4.12)$$

where l and n are the length of the source word E and the target word F, respectively.

To accelerate the convergence of EM training and reduce the number of the noisy TU aligned pairs (u_i, v_j), we restrict the combination of TU pairs to limited patterns. Basing on the assumption that the articulatory representations of phonemes are very similar across languages, the similarities of phonemes of TUs are classified based on phonetic knowledge. Consonant TU pairs only with the same or similar phonemes can be matched. An English consonant can also be matched with a Chinese syllable beginning with the same or similar phonemes. An English semivowel TU can either be matched with a Chinese consonant or with a vowel with the same or similar phonemes, or can be matched with a Chinese syllable beginning with the same or similar phonemes.

As for the probability $P(h,k)$, it is set to uniform distribution in the initialization phase, as shown in the following:

$$P(h,k) = \frac{1}{T}, \quad (4.13)$$

where T is the total number of match types allowed.

Based on the EM algorithm with Viterbi decoding (Forney, 1973), the iterative parameter estimation procedure is described as follows:

Step 1 (Initialization):

Use Eq. (4.12) to generate likely TU alignment pairs. Calculate the initial model parameters, $P(v_j \mid u_i)$ and $P(h,k)$, using Eq. (4.10) and Eq. (4.13), respectively.

Step 2 (Expectation):

Based on the current model parameters, find the best Viterbi path for each E and F word pair in the training set.

Step 3 (Maximization):

Based on all the TU alignment pairs obtained in Step 2, calculate the new model parameters using Eq. (4.10) and Eq. (4.11). Replace the model parameters with the new model parameters. If a stopping criterion or a predefined number of iterations is reached, then stop the training procedure. Otherwise, go back to Step 2.

In the first iteration, TUs in English and Chinese are constrained based on phonetic knowledge. However, in the subsequent iterations, the whole training process is run in a totally unsupervised manner. Therefore, some new TUs are

automatically discovered from the training data within the constraints of match types, as demonstrated in Chapter 7.

4.4 Alignment of transliteration pairs in parallel corpora

Machine transliteration is useful in many NLP applications, such as MT, CLIR, and bilingual lexicon construction. One interested problem is that of finding the transliteration equivalent for a given source word in a parallel corpus. In this section, we will introduce how the proposed transliteration model TM can be applied to perform this task. The task becomes more challenging for language pairs with different sound systems, such as Chinese/English, Japanese/English, and Arabic/English. Although we perform this task on the English-Chinese language pair, the proposed approach is easily extendable to other language pairs.

4.4.1 Overall Process

For the purpose of extracting name and transliteration pairs from parallel corpora, a sentence alignment procedure is applied first to align parallel texts at the sentence level. Then, we use a part of speech tagger to identify proper nouns in the English source sentence. After that, the machine transliteration model is applied to isolate the transliteration in the Chinese target sentence. In general, the proposed transliteration

model can be further augmented by linguistic processing, which will be described in more detail in the next subsection. The overall process is summarized in Figure 4.3.

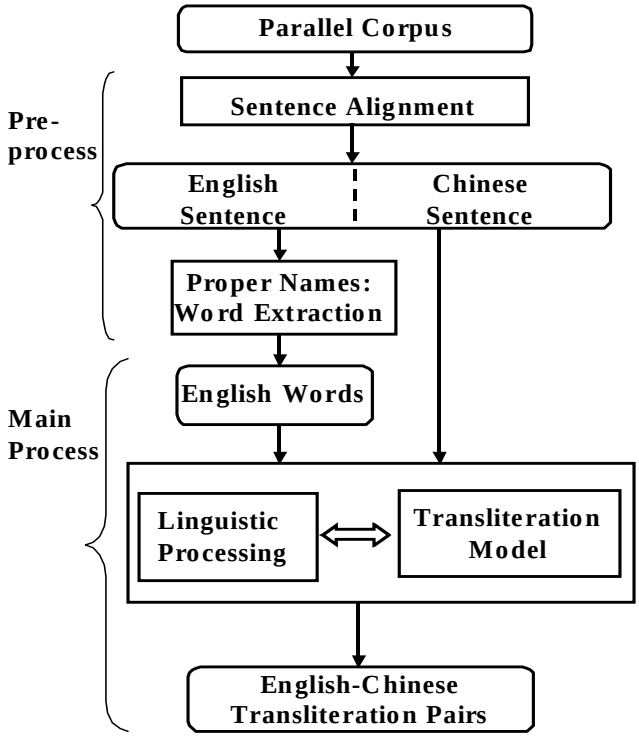

Figure 4.3 The overall process for extracting name and transliteration pairs from parallel corpora.

An excerpt from the magazine *Scientific American* (Cibelli et al., 2002) is given in the following:

Source sentence:

"Rudolf Jaenisch, a cloning expert at the Whitehead Institute for Biomedical Research at the Massachusetts Institute of Technology, concurred:"

Target sentence:

"麻省理工學院懷海德生物醫學研究院的複製專家傑尼西說："

In the above excerpt, three English proper nouns, “Jaenisch,” “Whitehead,” and “Massachusetts,” were identified from the results of tagging. Utilizing Eq. (4.6) and Viterbi decoding, we found that the target word “懷海德 (huaihaite)” most likely corresponded to “Whitehead.” The other word pair (Jaenisch, 傑尼西 “chiehnihsi”) can also be extracted through a similar process. However, the third word pair (Massachusetts, 麻省 “masheng”) failed to be extracted by the proposed approach. The reason is that “麻省” is an abbreviation of “麻薩諸塞州 (masachusaichou)” which is a well established popular translated name of “Massachusetts.” Therefore, the proposed model is incapable of resolving the abbreviation mentioned above.

In order to retrieve the transliteration for a given proper noun, we need to keep track of the optimal TU decoding sequence associated with the given Chinese term for each word pair under the proposed method. It can be easily obtained via backtracking the best Viterbi path (Manning and Schutze, 1999). For the name-transliteration pair (Whitehead, 懷海德) mentioned above, the alignments of the TU matching pairs via the Viterbi path are illustrated in Figure 4.4 and Figure 4.5.

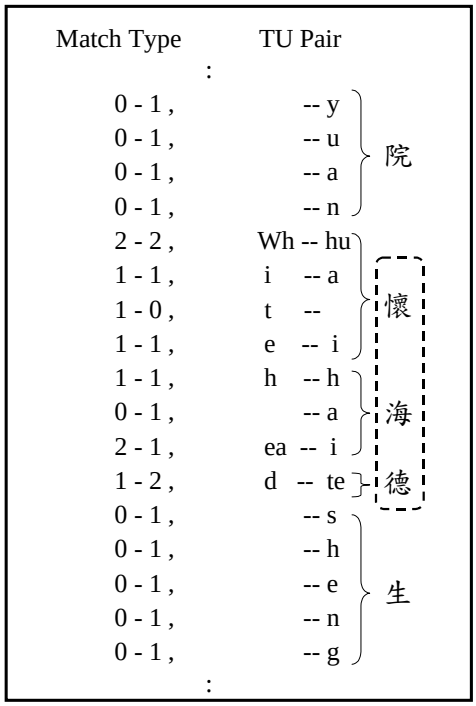

Figure 4.4 The alignments of the TU matching pairs via the Viterbi path.

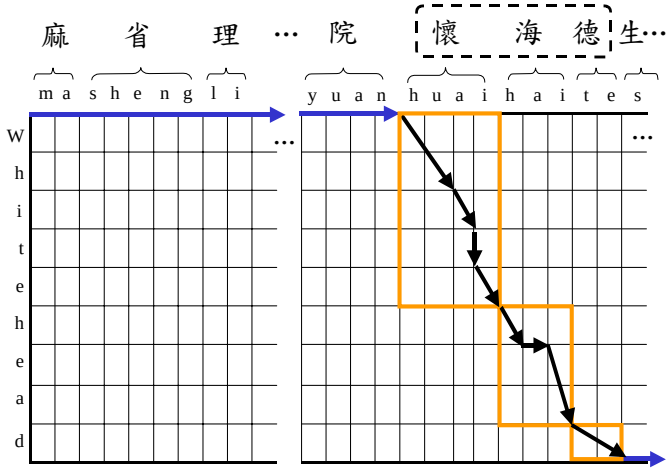

Figure 4.5 The Viterbi alignment path.

In this example, the word “Whitehead” is decomposed into seven TUs, “Wh-i-t-e-h-ea-d,” and aligned with the romanization “huaihaite” of the transliteration “懷海德.”

4.4.2 Linguistic Processing

Some language-dependent knowledge can be integrated to further improve the performance, especially when we focus on specific language pairs.

Linguistic Processing Rule 1 (R1)

Some source words have both transliterations and translations, which are equally acceptable and can be used interchangeably. For example, the translation and the transliteration of the source word "England" are "英國 (Yingkou)" and "英格蘭 (Yingkolan)," respectively, as shown in Figure 4.6. Since the proposed model is designed specifically for transliteration, such cases may cause problems. One way to overcome this limitation is to handle these cases by using a list of commonly used proper names and translations. A portion of the list is shown in Table 4.1.

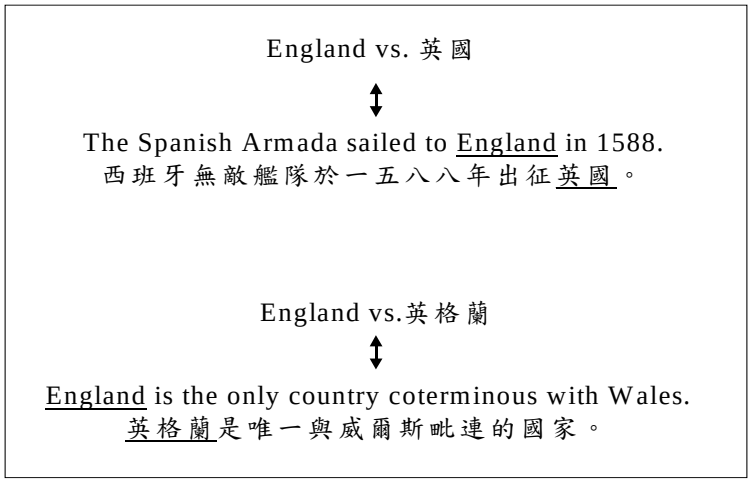

Figure 4.6 Examples of mixed usages of translation and transliteration.

Table 4.1 A portion of the list for translation.

Source Word	Target Word	Source Word	Target Word
Afghanistan	阿富汗	England	英國
America	美國	France	法國
Asia	亞洲	Greece	希臘
Canada	加拿大	India	印度
China	中國	Spanish	西班牙
Christ	耶穌	Yugoslavia	南斯拉夫

Linguistic Processing Rule 2 (R2)

From error analysis of the aligned results of the training set, we have found that the proposed approach suffers from fluid TUs, such as “t,” “d,” “tt,” “dd,” “te,” and “de.” Sometimes they are omitted during transliteration, and sometimes they are transliterated as Chinese characters. For instance, “d” is usually transliterated as “特,” “得,” or “德” corresponding to the Chinese TU of “te.” The English TU “d” is transliterated as “德” in (Clifford, 克利福德), but left out in (Radford, 雷德福). This phenomenon causes problems; in the example shown in Figure 4.7, the TU “d” in “David” is mistakenly matched up with “大衛的.”

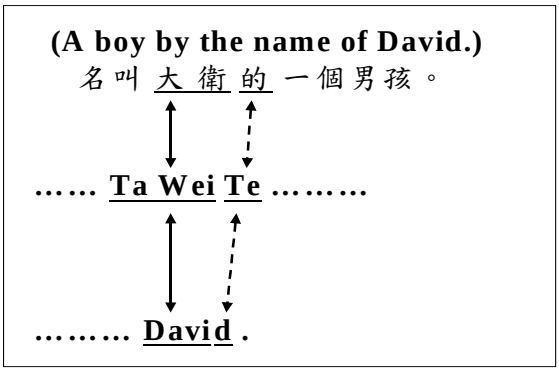

Figure 4.7 Example of transliterated word extraction for “David.”

Similarly, the English TU “s” or “se” is likely to misalign with “是” (TU “shih”) as in “雅典是古代希臘最強大的城邦之一。(Athens was one of the most powerful city-states of ancient Greece.).” See Figure 4.8 for more details.

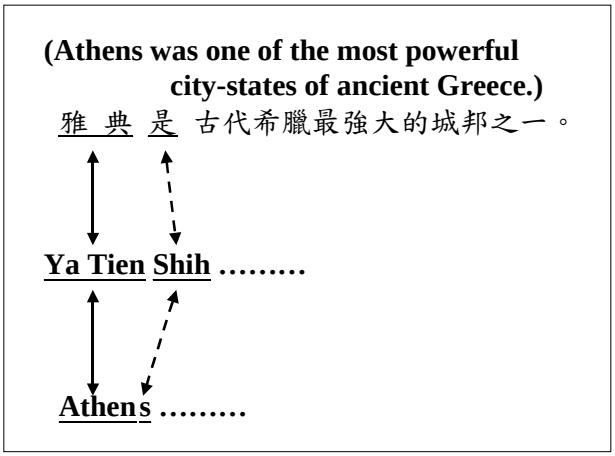

Figure 4.8 Example of transliterated word extraction for “Athens.”

However, the problem caused by fluid TUs can be partly overcome by adding more linguistic constraints in the post-processing phase. We calculate the Chinese character

distributions of proper nouns from the corpus. A small set of Chinese characters is often used for transliteration. Therefore, it is possible to improve the performance by pruning extra tailing characters, which do not belong to the transliterated character set, from the transliteration candidates. For instance, the probability of "的, 去, 說, 是, 有" being used in transliteration is very low. Therefore, the correct transliteration "大衛" for the source word "David" can be extracted by removing the character "的." We denote this strategy as Rule 2 (R2).

4.5 Work Flow of Integrating Linguistic and Statistical Information

Combining the linguistic processing and transliteration model, we present the algorithm for transliteration extraction as follows:

Step1: Look up the translation list as stated in R1. If the translation of a source word appears in both the entry of the translation list and the aligned target sentence (or paragraph), then pick the translation as the target word. Otherwise, go to Step 2.

Step 2: Pass the source word and its aligned target sentence (or paragraph) through the proposed model to extract the target word. Once this is done, go to Step 3.

Step 3: Apply linguistic processing R2 to remove superfluous tailing characters from the extracted transliterations.

After the above steps are completed, the performance of source-target word extraction is significantly improved. The experimental results will be described in more detail in Chapter 7.