Machine Transliteration using Target-Language Grapheme and Phoneme: Multi-engine Transliteration Approach

Jong-Hoon Oh, Kiyotaka Uchimoto, and Kentaro Torisawa
Language Infrastructure Group, MASTAR Project,
National Institute of Information and Communications Technology (NICT)
3-5 Hikaridai Seika-cho, Soraku-gun, Kyoto 619-0289 Japan
{rovellia,uchimoto,torisawa}@nict.go.jp

Abstract

This paper describes our approach to "NEWS 2009 Machine Transliteration Shared Task." We built multiple transliteration engines based on different combinations of two transliteration models and three machine learning algorithms. Then, the outputs from these transliteration engines were combined using re-ranking functions. Our method was applied to all language pairs in "NEWS 2009 Machine Transliteration Shared Task." The official results of our standard runs were ranked the best for four language pairs and the second best for three language pairs.

1 Outline

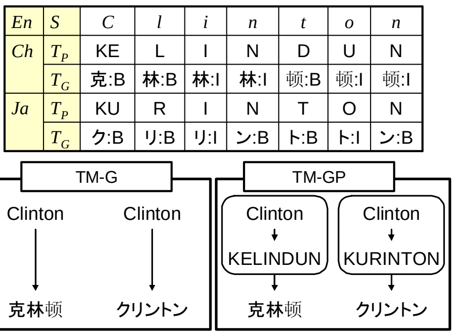
This paper describes our approach to "NEWS 2009 Machine Transliteration Shared Task." Our approach was based on two transliteration models – **TM-G** (**T**ransliteration **m**odel based on target-language **G**raphemes) and **TM-GP** (**T**ransliteration **m**odel based on target-language **G**raphemes and **P**honemes). The difference between the two models lies in whether or not a machine transliteration process depends on target-language phonemes. TM-G directly converts source-language graphemes into target-language graphemes, while TM-GP first transforms source language graphemes into target-language phonemes and then target-language phonemes coupled with their corresponding source-language graphemes are converted into target-language graphemes. We used three different machine learning algorithms (conditional random fields (CRFs), margin infused relaxed algorithm (MIRA), and maximum entropy model (MEM)) (Berger et al., 1996; Crammer and Singer, 2003; Lafferty et al., 2001) for building multiple machine transliteration engines. We attempted to improve the transliteration quality by combining the outputs of different machine transliteration engines operating on the same input. Our approach was applied to all language pairs in "NEWS 2009 Machine Transliteration Shared Task." The official results of our approach were ranked as the best for four language pairs and the second best for three language pairs (Li et al., 2009a).

2 Transliteration Model

Let **S** be a source-language word and **T** be a target-language transliteration of **S**. **T** is represented in two ways – T_G, a sequence of target-language graphemes, and T_P, a sequence of target-language phonemes. Here, a target-language grapheme is defined as a target-language character. We regard consonant and vowel parts in the romanized form of a target language grapheme as a target-language phoneme. Then **TM-G** and **TM-GP** are formulated as Eq (1) and (2), respectively.

$$P_{TM-G}(T|S) = P(T_G|S) \tag{1}$$

$$P_{TM-GP}(T|S) \tag{2}$$
$$= \sum_{\forall T_P} P(T_P|S) \times P(T_G|T_P, S)$$



En	*S*	*C*	*l*	*i*	*n*	*t*	*o*	*n*
Ch	T_P	KE	L	I	N	D	U	N
	T_G	克:B	林:B	林:I	林:I	顿:B	顿:I	顿:I
Ja	T_P	KU	R	I	N	T	O	N
	T_G	ク:B	リ:B	リ:I	ン:B	ト:B	ト:I	ン:B

Figure 1: Illustration of the two transliteration models

Figure 1 illustrates the two transliteration models with examples, *Clinton* and its Chinese and Japanese transliterations. Target language graphemes are represented in terms of the BIO notation. This makes it easier to represent many-to-one correspondence between target language phoneme and grapheme.

3 Machine Learning Algorithms

A machine transliteration problem can be converted into a sequential labeling problem, where each source-language grapheme is tagged with its corresponding target-language grapheme. This section briefly describes the machine learning algorithms used for building multiple transliteration engines.

3.1 Maximum Entropy Model

Machine transliteration based on the maximum entropy model was described in detail in Oh et al. (2006) along with comprehensive evaluation of its performance. We used the same way as that proposed by Oh et al. (2006), thus its full description is not presented here.

3.2 Conditional Random Fields (CRFs)

CRFs, a statistical sequence modeling framework, was first introduced by Lafferty et al. (2001). CRFs has been used for sequential labeling problems such as text chunking and named entity recognition (McCallum and Li, 2003). CRF++[1] was used in our experiment.

3.3 Margin Infused Relaxed Algorithm

The Margin Infused Relaxed Algorithm (MIRA) has been introduced by Crammer and Singer (2003) for large-margin multi-class classification. Kruengkrai et al. (2008) proposed a discriminative model for joint Chinese segmentation and POS tagging, where MIRA was used as their machine learning algorithm. We used the same model for our machine transliteration, exactly joint syllabication[2] and transliteration.

3.4 Features

We used the following features within the ± 3 context window[3] for the above mentioned three machine learning algorithms.

- Left-three and right-three source-language graphemes (or syllables)
- Left-three and right-three target-language phonemes
- Target-language graphemes assigned to the previous three source-language graphemes (or syllables)

[1]Available at `http://crfpp.sourceforge.net/`

[2]A syllable in English is defined as a sequence of English grapheme corresponding to one target-language grapheme.

[3]The unit of context window is source-language grapheme or syllable.

4 Multi-engine Transliteration

4.1 Individual Transliteration Engine

The main aim of the multi-engine transliteration approach is to combine the outputs of multiple engines so that the final output is better in quality than the output of each individual engine. We designed four transliteration engines using different combinations of source-language transliteration units, transliteration models, and machine learning algorithms as listed in Table 1. We named four transliteration engines as CRF-G, MEM-G, MEM-GP, and MIRA-G. Here, the prefixes represent applied machine learning algorithms (maximum entropy model (MEM), CRFs, and MIRA), while G and GP in the suffix represent the transliteration models, **TM-G** and **TM-GP**, respectively. Each individual engine produces 30-best transliterations for a given source-language word.

	Source-language transliteration unit	
	Grapheme	Syllable
TM-G	ME-G, CRF-G	MIRA-G
TM-GP	ME-GP	N/A

Table 1: Design strategy for multiple transliteration engines

4.2 Combining Methodology

We combined the outputs of multiple transliteration engines by means of a re-ranking function, $g(x)$. Let X be a set of transliterations generated by multiple transliteration engines for source-language word s and ref be a reference transliteration of s. A re-ranking function is defined as Eq. (3), where it ranks ref in X higher and the others lower (Oh and Isahara, 2007).

$$g(x) : X \rightarrow \{r : r\ is\ ordering\ of\ x \in X\} \quad (3)$$

We designed two types of re-ranking functions by using the rank of each individual engine and machine learning algorithm.

4.2.1 Re-ranking Based on the Rank of Individual Engines

Two re-ranking functions based on the rank of each individual engine, g_{rank} and $g_{Fscore}(x)$, are used for combining the outputs of multiple transliteration engines. Let X be a set of outputs of N transliteration engines for the same input. $g_{rank}(x)$ re-ranks $x \in X$ in the manner shown in Eq. (4), where $Rank_i(x)$ is the position of x in the n-best list generated by the i^{th} transliteration engine. $g_{rank}(x)$ can be interpreted as the average rank of x over outputs of each individual engine. If x is not in the n-best list of the i^{th} transliteration engine, $\frac{1}{Rank_i(x)} = 0$.

$$g_{rank}(x) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Rank_i(x)} \tag{4}$$

$g_{Fscore}(x)$ is based on $g_{rank}(x)$ and the F-score measure, which is one of the evaluation metrics in the "NEWS 2009 Machine Transliteration Shared Task" (Li et al., 2009b). We considered the top three outputs of each individual engine as reference transliterations and defined them as *virtual reference transliterations*. We calculated the F-score measure between the virtual reference transliteration and each output of multiple transliteration engines. $g_{Fscore}(x)$ is defined by Eq. (5), where VRef is a set of virtual reference transliterations, and $F_{score}(vr, x)$ is a function that restores the F-score measure between vr and x.

$$g_{Fscore}(x) = g_{rank}(x) \times MF(x) \tag{5}$$
$$MF(x) = \frac{1}{|VRef|}\sum_{vr\in VRef} F_{score}(vr, x)$$

Since the F-score measure is calculated in terms of string similarity, x gets a high score from $g_{MF}(x)$ when it is orthographically similar to virtual reference transliterations.

4.2.2 Re-ranking based on Machine Learning Algorithm

We used the maximum entropy model for learning re-ranking function $g_{ME}(x)$. Let ref be a reference transliteration of source-language word s, $feature(x)$ be a feature vector of $x \in X$, and $y \in \{ref, wrong\}$ be the training label for x. $g_{ME}(x)$ assigns a probability to $x \in X$ as shown in Eq. (6).

$$g_{ME}(x) = P(ref|feature(x)) \tag{6}$$

A feature vector of x is composed of

- $\langle g_{rank}(x), g_{Fscore}(x), \frac{1}{Rank_i(x)}, P(T|S)\rangle$

where $\frac{1}{Rank_i(x)}$ and $P(T|S)$ of each individual engine are used as a feature.

We estimated $P(ref|feature(x))$ by using the development data.

5 Our Results

5.1 Individual Engine

	CRF-G	MEM-G	MEM-GP	MIRA-G
EnCh	0.628	0.686	**0.715**	0.684
EnHi	0.455	**0.469**	**0.469**	0.412
EnJa	0.514	0.517	**0.519**	0.490
EnKa	**0.386**	0.380	0.380	0.338
EnKo	**0.460**	0.438	0.447	0.367
EnRu	**0.600**	0.561	0.566	0.568
EnTa	0.453	**0.459**	**0.459**	0.412
JnJk	N/A	<u>0.532</u>	N/A	**0.571**

Table 2: ACC of individual engines on the test data

Table 2 presents ACC[4] of individual transliteration engines, which was applied to all language pairs in "NEWS 2009 Machine Transliteration Shared Task" (Li et al., 2004; Kumaran and Kellner, 2007; The CJK Dictionary Institute, 2009). CRF-G was the best transliteration engine in EnKa, EnKo, and EnRu. Owing to the high training costs of CRFs, we trained CRF-G in EnCh with a very small number of iterations[5]. Hence, the performance of CRF-G was poorer than that of the other engines in EnCh. MEM-GP was the best transliteration engine in EnCh, EnHi, EnJa, and EnTa. These results indicate that joint use of source language graphemes and target language phonemes were very useful for improving performance. MIRA-G was sensitive to the training data size, because it was based on joint syllabication and transliteration. Therefore, the performance of MIRA-G was relatively better in EnCh and EnJa, whose training data size is bigger than other language pairs. CRF-G could not be applied to JnJk, mainly due to too long training time. Further, MEM-GP could not be applied to JnJk, because transliteration in JnJk can be regarded as conversion of target language phonemes to target language graphemes. MEM-G and MIRA-G were

[4]Word accuracy in Top-1 (Li et al., 2009b)

[5]We applied over 100 iterations to other language pairs but only 30 iterations to EnCh.

applied to JnJk and MIRA-G showed the best performance in JnJK.[6]

5.2 Combining Multiple Engines

	g_{rank}	g_{Fscore}	g_{ME}	I-BEST
EnCh	0.730	**0.731**	$\underline{\mathbf{0.731}}$	0.715
EnHi	0.481	0.475	$\underline{\mathbf{0.483}}$	0.469
EnJa	0.535	0.535	$\underline{\mathbf{0.537}}$	0.519
EnKa	0.393	**0.399**	$\underline{0.398}$	0.386
EnKo	0.461	0.444	$\underline{\mathbf{0.473}}$	0.460
EnRu	0.602	$\underline{\mathbf{0.605}}$	0.600	0.600
EnTa	0.470	**0.478**	$\underline{0.474}$	0.459
JnJk	**0.597**	0.593	0.590	0.571

Table 3: Multi-engine transliteration results on the test data: the underlined figures are our official result

Table 3 presents the ACC of our multi-engine transliteration approach and that of the best individual engine (I-BEST) in each language pair. g_{ME} gave the best performance in EnCh, EnHi, EnJa, and EnKo, while g_{Fscore} did in EnCh, EnKa, EnRu, and EnTa. Comparison between the best individual transliteration engine and our multi-engine transliteration showed that g_{rank} and g_{ME} consistently showed better performance except in EnRu, while g_{Fscore} showed the poorer performance in EnKo. The results to be submitted as "the standard run" were selected among the results listed in Table 3 by using cross-validation on the development data. We submitted the results of g_{ME} as the standard run to "NEWS 2009 Machine Transliteration Shared Task" for the six language pairs in Table 3, while the result of g_{Fscore} is submitted as the standard run for EnRu. The official results of our standard runs were ranked the best for EnCh, EnJa, EnKa, and EnTa, and the second best for EnHi, EnKo, and EnRu (Li et al., 2009a).

6 Conclusion

In conclusion, we have applied multi-engine transliteration approach to "NEWS 2009 Machine Transliteration Shared Task." We built multiple transliteration engines based on different combinations of transliteration models and machine learning algorithms. We showed that the transliteration model, which is based on target language graphemes and phonemes, and our multi-engine transliteration approach are effective, regardless of the nature of the language pairs.

[6]We submitted the results of MEM-G as a standard run for JnJk because we had only one transliteration engine for JnJK before the submission deadline of the NEWS 2009 machine transliteration shared task.

References

A. L. Berger, S. D. Pietra, and V. J. D. Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Canasai Kruengkrai, Jun'ichi Kazama, Kiyotaka Uchimoto, Kentaro Torisawa, and Hitoshi Isahara. 2008. A discriminative hybrid model for joint Chinese word segmentation and pos tagging. In *Proc. of The 11th Oriental COCOSDA Workshop*.

A. Kumaran and Tobias Kellner. 2007. A generic framework for machine transliteration. In *Proc. of SIGIR '07*, pages 721–722.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML01*, pages 282–289.

Haizhou Li, Min Zhang, and Su Jian. 2004. A joint source-channel model for machine transliteration. In *Proc. of ACL '04*, pages 160–167.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009a. Report on NEWS 2009 machine transliteration shared task. In *Proc. of ACL-IJCNLP 2009 Named Entities Workshop*.

Haizhou Li, A Kumaran, Min Zhang, and Vladimir Pervouchine. 2009b. Whitepaper of NEWS 2009 machine transliteration shared task. In *Proc. of ACL-IJCNLP 2009 Named Entities Workshop*.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proc. of CoNLL '03*, pages 188–191.

Jong-Hoon Oh and Hitoshi Isahara. 2007. Machine transliteration using multiple transliteration engines and hypothesis re-ranking. In *Proc. of the 11th Machine Translation Summit*, pages 353–360.

Jong-Hoon Oh, Key-Sun Choi, and Hitoshi Isahara. 2006. A comparison of different machine transliteration models. *Journal of Artificial Intelligence Research (JAIR)*, 27:119–151.

The CJK Dictionary Institute. 2009. `http://www.cjk.org`.