Fast decoding and Easy Implementation: Transliteration as Sequential Labeling

Eiji ARAMAKI
The University of Tokyo
eiji.aramaki@gmail.com

Takeshi ABEKAWWA
National Institute of Informatics
abekawa@nii.ac.jp

Abstract

Although most of previous transliteration methods are based on a generative model, this paper presents a discriminative transliteration model using conditional random fields. We regard character(s) as a kind of label, which enables us to consider a transliteration process as a sequential labeling process. This approach has two advantages: (1) fast decoding and (2) easy implementation. Experimental results yielded competitive performance, demonstrating the feasibility of the proposed approach.

1 Introduction

To date, most transliteration methods have relied on a generative model which resembles a statistical machine translation (SMT) model. Although the generative approach has appealing feasibility, it usually suffers from parameter settings, length biases and decoding time.

We assume a transliteration process as a kind of sequential labeling that is widely employed for various tasks, such as Named Entity Recognition (NER), part-of-speech (POS) labeling, and so on. Figure 1 shows a lattice of both the transliteration and POS labeling. As shown in that figure, both tasks share a similar work frame: (1) an input sequence is decomposed into several segments; then (2) each segments produces a label. Although the label represents a POS in POS labeling, it represents a character (or a character sequence) in the transliteration task.

The proposed approach entails three risks.

1. **Numerous Label Variation**: Although POS requires only 10–20 labels at most, a transliteration process requires numerous labels. In fact, Japanese katakana requires more than 260 labels in the following experiment (we consider combinations of characters as a label). Such a huge label set might require extremely heavy calculation.

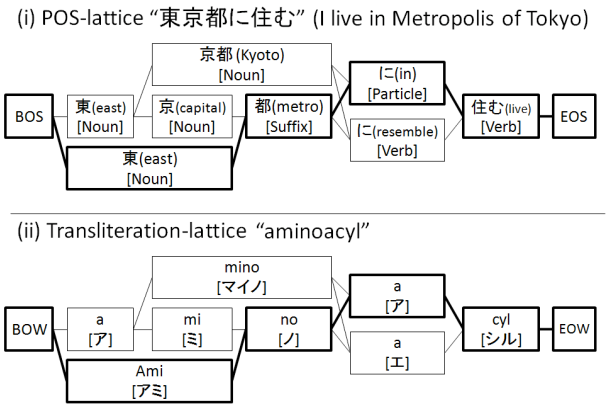

Figure 1: (i) Part-of-Speech Lattice and (ii) Transliteration Lattice.

2. **No Gold Standard Data**: We build the gold standard label from character alignment using GIZA++ [1]. Of course, such gold standard data contain alignment errors, which might decrease labeling performance.

3. **No Language Model**: The proposed approach cannot incorporate the target language model.

In spite of the disadvantages listed above, the proposed method offers two strong advantages.

1. **Fast Decoding**: Decoding (more precisely labeling) is extremely fast (0.12–0.58 s/input). Such rapid decoding is useful for various applications, for example, a query expansion for a search engine and so on [2].

[1] http://www.fjoch.com/GIZA++.html

[2] A fast transliteration demonstration is available at the web site; http://akebia.hcc.h.u-tokyo.ac.jp/NEWS/

Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 65–68,
Suntec, Singapore, 7 August 2009.

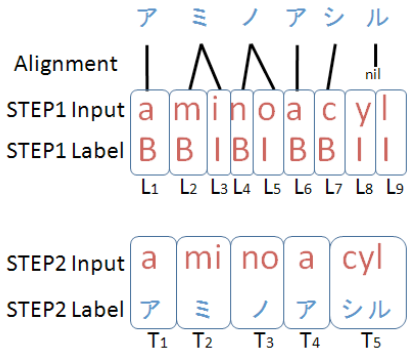

Figure 2: Conversion from Training set to Gold Standard Labels

2. **Easy Implementation**: Because sequential labeling is a traditional research topic, various algorithms and tools are available. Using them, we can easily realize various transliteration systems in any language pairs.

The experimental results empirically demonstrate that the proposed method is competitive in several language directions (e.g. English–Chinese).

2 Method

We developed a two-stage labeling system. First, an input term is decomposed into several segments (STEP1). Next, each segmentation produces symbol(s) (STEP2).

2.1 STEP1: Chunking

For a given noun phrase, consisting n characters, the system gave a label ($L_1 ... L_n$) that represents segmentations.

The segmentation is expressed as two types of labels (label B and I), where B signifies a beginning of the segmentation, and I signifies the end of segmentation. This representation is similar to the IOB representation, which is used in Named Entity Recognition (NER) or chunking.

For label prediction, we used Conditional Random Fields (CRFs), which is a state-of-the-art labeling algorithm. We regard a source character itself as a CRF feature. The window size is three (the current character and previous/next character).

2.2 STEP2: Symbol production

Next, the system estimates labels ($T_1 ... T_m$) for each segmentation, where m is the number of segmentations (the number of B labels in STEP1). The label of this step directly represents a target language character(s). The method of building a gold standard label is described in the next subsection.

Like STEP1, we use CRFs, and regard source characters as a feature (window size=3).

Table 1: Corpora and Sizes

Notation	Language	Train	Test
EN-CH	English–Chinese	31,961	2,896
EN-JA	English–Japanese	27,993	1,489
EN-KO	English–Korean	4,840	989
EN-HI	English–Hindi	10,014	1,000
EN-TA	English–Tamil	8,037	1,000
EN-KA	English–Kannada	8,065	1,000
EN-RU	English–Russian	5,977	1,000

* EN-CH is provided by (Li et al., 2004); EN-TA, EN-KA, EN-HI and EN-RU are from (Kumaran and Kellner, 2007); EN-JA and EN-KO are from `http://www.cjk.org/`.

2.3 Conversion from Alignment to Labels

First, character alignment is estimated using GIZA++ as shown at the top of Fig. 2. The alignment direction is a target- language-to-English, assuming that n English characters correspond to a target language character.

The STEP1 label is generated for each English character. If the alignment is 1:1, we give the character a B label. If the alignment is $n:1$, we assign the first character a B label, and give the others I. Note that we regard null alignment as a continuance of the last segmentation (I).

The STEP2 label is generated for each English segmentation (B or $BI*$). If a segmentation corresponds to two or more characters in the target side, we regard the entire sequence as a label (see T_5 in Fig. 2).

3 Experiments

3.1 Corpus, Evaluation, and Setting

To evaluate the performance of our system, we used a training-set and test-set provided by NEWS[3](Table 1).

We used the following six metrics (Table 2) using 10 output candidates. A white paper[4] presents the detailed definitions. For learning, we used CRF++[5] with standard parameters (f=20, c=.5).

[3]http://www.acl-ijcnlp-2009.org/workshops/NEWS2009/
[4]https://translit.i2r.a-star.edu.sg/news2009/whitepaper/
[5]http://crfpp.sourceforge.net/

Table 3: Results in Test-set

	ACC	Mean$_F$	MRR	MAP$_{ref}$	MAP$_{10}$	MAP$_{sys}$
EN–CH	0.580	0.826	0.653	0.580	0.199	0.199
EN–RU	0.531	0.912	0.635	0.531	0.219	0.219
EN–JA	0.457	0.828	0.576	0.445	0.194	0.194
EN–TA	0.365	0.884	0.504	0.360	0.172	0.172
EN–HI	0.363	0.864	0.503	0.360	0.170	0.170
EN–KA	0.324	0.856	0.438	0.315	0.148	0.148
EN–KO	0.170	0.512	0.218	0.170	0.069	0.069

Table 2: Evaluation Metrics

ACC	Word Accuracy in Top 1.
Mean$_F$	The mean$_F$ measures the fuzzy accuracy that is defined by the edit distance and Longest Common Subsequence (LCS).
MRR	Mean Reciprocal Rank. 1/MRR tells approximately the average rank of the correct transliteration.
MAP$_{ref}$	Measures the precision in the $n-$best candidates tightly for each reference.
MAP$_{10}$	Measures the precision in the 10-best candidates.
MAP$_{sys}$	Measures the precision in the top K_i-best candidates produced by the system.

3.2 Results and Discussion

Table 3 presents the performance. As shown in the table, a significant difference was found between languages (from low (0.17) to high (0.58)).

The high accuracy results(EN-CH or EN-RU) are competitive with other systems (the middle rank among the NEWS participating systems). However, several language results (such as EN-KO) were found to have poor performance.

We investigated the difference between high-performance languages and the others. Table 4 shows the training/test times and the number of labels. As shown in the table, wide divergence is apparent in the number of labels. For example, although EN–KO requires numerous labels (536 labels), EN–RU needs only 131 labels. This divergence roughly corresponds to both training-time and accuracy as follows: (1) EN–KO requires long training time (11 minutes) which gave poor performance (0.17 ACC), and (2) EN–RU requires short training (only 26.3 seconds) which gave high performance (0.53 ACC). This suggests that if the number of labels is small, we successfully convert transliteration into a sequential labeling task.

The test time seemed to have no relation to both training time and performance. To investigate what gave effects on test time is a subject for our future work.

Table 4: Average Test time, Training Time, and the number of labels (label variation).

Language	Test	Train	# of labels
EN–KO	0.436s	11m09.5s	536
EN–CH	0.201s	6m18.9s	283
EN–JA	0.247s	4m44.3s	269
EN–KA	0.190s	2m26.6s	231
EN–HI	0.302s	1m55.6s	268
EN–TA	0.124s	1m32.9s	207
EN–RU	0.580s	0m26.3s	131

* Test time is the average labeling time for an input. Training time is the average training time for 1000 labels.

4 Related Works

Most previous transliteration studies have relied on a generative model resembling the IBM model(Brown et al., 1993). This approach is applicable to various languages: for Japanese (Goto et al., 2004; Knight and Graehl, 1998), Korean(Oh and Choi, 2002; Oh and Choi, 2005; Oh and Isahara, 2007), Arabic(Stalls and Knight, 1998; Sherif and Kondrak, 2007), Chinese(Li et al., 2007), and Persian(Karimi et al., 2007). As described previously, the proposed discriminative approach differs from them.

Another perspective is that of how to represent transliteration phenomena. Methods can be classified into three main types: (1) grapheme-based (Li et al., 2004), (2) phoneme-based (Knight and Graehl, 1998), and (3) combinations of these methods (hybrid-model(Bilac and Tanaka, 2004), and a correspondence-based model(Oh and Choi, 2002; Oh and Choi, 2005) re-ranking model (Oh and Isahara, 2007)). Our proposed method employs a grapheme-based approach. Employing phonemes is a challenge reserved for future studies.

Aramaki et al. (2008) proposed a discrimina-

tive transliteration approach using Support Vector Machines (SVMs). However, their goal, which is to judge whether two terms come from the same English words or not, differs from this paper goal.

5 Conclusions

This paper presents a discriminative transliteration model using a sequential labeling technique. Experimental results yielded competitive performance, demonstrating the feasibility of the proposed approach. In the future, how to incorporate more rich information, such as language model and phoneme, is remaining problem. We believe this task conversion, from generation to sequential labeling, can be useful for several practical applications.

ACKNOWLEDGMENT

Part of this research is supported by Japanese Grant-in-Aid for Scientific Research (A) Number:20680006.

References

Eiji Aramaki, Takeshi Imai, Kengo Miyo, and Kazuhiko Ohe. 2008. Orthographic disambiguation incorporating transliterated probability. In *Proceedings of International Joint Conference on Natural Language Processing (IJCNLP2008)*, pages 48–55.

Slaven Bilac and Hozumi Tanaka. 2004. A hybrid back-transliteration system for Japanese. In *Proceedings of The 20th International Conference on Computational Linguistics (COLING2004)*, pages 597–603.

Peter F. Brown, Stephen A. Della Pietra, Vi cent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2).

Isao Goto, Naoto Kato, Terumasa Ehara, and Hideki Tanaka. 2004. Back transliteration from Japanese to English using target English context. In *Proceedings of The 20th International Conference on Computational Linguistics (COLING2004)*, pages 827–833.

Sarvnaz Karimi, Falk Scholer, and Andrew Turpin. 2007. Collapsed consonant and vowel models: New approaches for English-Persian transliteration and back-transliteration. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL2007)*, pages 648–655.

Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612.

A. Kumaran and Tobias Kellner. 2007. A generic framework for machine transliteration. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 721–722.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the Meeting of the Association for Computational Linguistics (ACL2004)*, pages 159–166.

Haizhou Li, Khe Chai Sim, Jin-Shea Kuo, and Minghui Dong. 2007. Semantic transliteration of personal names. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL2007)*, pages 120–127.

Jong-Hoon Oh and Key-Sun Choi. 2002. An English-Korean transliteration model using pronunciation and contextual rules. In *Proceedings of The 19th International Conference on Computational Linguistics (COLING2002)*, pages 758–764.

Jong-Hoon Oh and Key-Sun Choi. 2005. An ensemble of grapheme and phoneme for machine transliteration. In *Proceedings of Second International Joint Conference on Natural Language Processing (IJCNLP2005)*, pages 450–461.

Jong-Hoon Oh and Hitoshi Isahara. 2007. Machine transliteration using multiple transliteration engines and hypothesis re-ranking. In *Proceedings of MT Summit XI*, pages 353–360.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-based transliteration. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL2007)*, pages 944–951.

Bonnie Glover Stalls and Kevin Knight. 1998. Translating names and technical terms in arabic text. In *Proceedings of The International Conference on Computational Linguistics and the 36th Annual Meeting of the Association of Computational Linguistics (COLING-ACL1998) Workshop on Computational Approaches to Semitic Languages*.