Jointly optimizing a two-step conditional random field model for machine transliteration and its fast decoding algorithm

Dong Yang, Paul Dixon and Sadaoki Furui
Department of Computer Science
Tokyo Institute of Technology
Tokyo 152-8552 Japan
{raymond,dixonp,furui}@furui.cs.titech.ac.jp

Abstract

This paper presents a joint optimization method of a two-step conditional random field (CRF) model for machine transliteration and a fast decoding algorithm for the proposed method. Our method lies in the category of direct orthographical mapping (DOM) between two languages without using any intermediate phonemic mapping. In the two-step CRF model, the first CRF segments an input word into chunks and the second one converts each chunk into one unit in the target language. In this paper, we propose a method to jointly optimize the two-step CRFs and also a fast algorithm to realize it. Our experiments show that the proposed method outperforms the well-known joint source channel model (JSCM) and our proposed fast algorithm decreases the decoding time significantly. Furthermore, combination of the proposed method and the JSCM gives further improvement, which outperforms state-of-the-art results in terms of top-1 accuracy.

1 Introduction

There are more than 6000 languages in the world and 10 languages of them have more than 100 million native speakers. With the information revolution and globalization, systems that support multiple language processing and spoken language translation become urgent demands. The translation of named entities from alphabetic to syllabary language is usually performed through transliteration, which tries to preserve the pronunciation in the original language.

For example, in Chinese, foreign words are written with Chinese characters; in Japanese, foreign words are usually written with special characters called Katakana; examples are given in Figure 1.

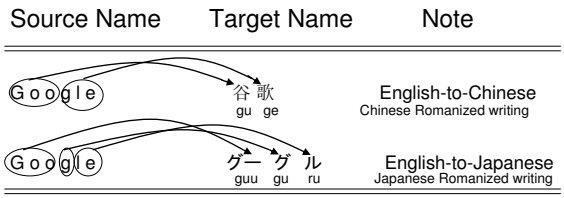

Figure 1: Transliteration examples

An intuitive transliteration method (Knight and Graehl, 1998; Oh et al., 2006) is to firstly convert a source word into phonemes, then find the corresponding phonemes in the target language, and finally convert them to the target language's written system. There are two reasons why this method does not work well: first, the named entities have diverse origins and this makes the grapheme-to-phoneme conversion very difficult; second, the transliteration is usually not only determined by the pronunciation, but also affected by how they are written in the original language.

Direct orthographical mapping (DOM), which performs the transliteration between two languages directly without using any intermediate phonemic mapping, is recently gaining more attention in the transliteration research community, and it is also the "Standard Run" of the "NEWS 2009 Machine Transliteration Shared Task" (Li et al., 2009). In this paper, we try to make our system satisfy the standard evaluation condition, which requires that the system uses the provided parallel corpus (without pronunciation) only, and cannot use any other bilingual or monolingual resources.

The source channel and joint source channel models (JSCMs) (Li et al., 2004) have been proposed for DOM, which try to model $P(T|S)$ and $P(T,S)$ respectively, where T and S denote the words in the target and source languages. Ekbal et al. (2006) modified the JSCM to incorporate different context information into the model for

Proceedings of the ACL 2010 Conference Short Papers, pages 275–280,
Uppsala, Sweden, 11-16 July 2010.

Indian languages. In the "NEWS 2009 Machine Transliteration Shared Task", a new two-step CRF model for transliteration task has been proposed (Yang et al., 2009), in which the first step is to segment a word in the source language into character chunks and the second step is to perform a context-dependent mapping from each chunk into one written unit in the target language.

In this paper, we propose to jointly optimize a two-step CRF model. We also propose a fast decoding algorithm to speed up the joint search. The rest of this paper is organized as follows: Section 2 explains the two-step CRF method, followed by Section 3 which describes our joint optimization method and its fast decoding algorithm; Section 4 introduces a rapid implementation of a JSCM system in the weighted finite state transducer (WFST) framework; and the last section reports the experimental results and conclusions. Although our method is language independent, we use an English-to-Chinese transliteration task in all the explanations and experiments.

2 Two-step CRF method

2.1 CRF introduction

A chain-CRF (Lafferty et al., 2001) is an undirected graphical model which assigns a probability to a label sequence $L = l_1 l_2 \dots l_T$, given an input sequence $C = c_1 c_2 \dots c_T$. CRF training is usually performed through the L-BFGS algorithm (Wallach, 2002) and decoding is performed by the Viterbi algorithm. We formalize machine transliteration as a CRF tagging problem, as shown in Figure 2.

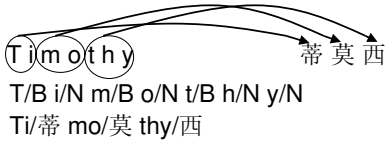

Figure 2: An pictorial description of a CRF segmenter and a CRF converter

2.2 CRF segmenter

In the CRF, a feature function describes a co-occurrence relation, and it is usually a binary function, taking the value 1 when both an observation and a label transition are observed. Yang et al. (2009) used the following features in the segmentation tool:

- Single unit features: $C_{-2}, C_{-1}, C_0, C_1, C_2$
- Combination features: $C_{-1}C_0, C_0C_1$

Here, C_0 is the current character, C_{-1} and C_1 denote the previous and next characters, and C_{-2} and C_2 are the characters located two positions to the left and right of C_0.

One limitation of their work is that only top-1 segmentation is output to the following CRF converter.

2.3 CRF converter

Similar to the CRF segmenter, the CRF converter has the format shown in Figure 2.

For this CRF, Yang et al. (2009) used the following features:

- Single unit features: CK_{-1}, CK_0, CK_1
- Combination features: $CK_{-1}CK_0$, CK_0CK_1

where CK represents the source language chunk, and the subscript notation is the same as the CRF segmenter.

3 Joint optimization and its fast decoding algorithm

3.1 Joint optimization

We denote a word in the source language by S, a segmentation of S by A, and a word in the target langauge by T. Our goal is to find the best word $\hat{T}$ in the target language which maximizes the probability $P(T|S)$.

Yang et al. (2009) used only the best segmentation in the first CRF and the best output in the second CRF, which is equivalent to

$$\begin{aligned} \hat{A} &= \arg\max_{A} P(A|S) \\ \hat{T} &= \arg\max_{T} P(T|S, \hat{A}), \end{aligned} \quad (1)$$

where $P(A|S)$ and $P(T|S, A)$ represent two CRFs respectively. This method considers the segmentation and the conversion as two independent steps. A major limitation is that, if the segmentation from the first step is wrong, the error propagates to the second step, and the error is very difficult to recover.

In this paper, we propose a new method to jointly optimize the two-step CRF, which can be

written as:

$$\begin{aligned}\hat{T} &= \arg\max_T P(T|S) \\ &= \arg\max_T \sum_A P(T, A|S) \\ &= \arg\max_T \sum_A P(A|S)P(T|S, A)\end{aligned} \quad (2)$$

The joint optimization considers all the segmentation possibilities and sums the probability over all the alternative segmentations which generate the same output. It considers the segmentation and conversion in a unified framework and is robust to segmentation errors.

3.2 N-best approximation

In the process of finding the best output using Equation 2, a dynamic programming algorithm for joint decoding of the segmentation and conversion is possible, but the implementation becomes very complicated. Another direction is to divide the decoding into two steps of segmentation and conversion, which is this paper's method. However, exact inference by listing all possible candidates explicitly and summing over all possible segmentations is intractable, because of the exponential computation complexity with the source word's increasing length.

In the segmentation step, the number of possible segmentations is 2^N, where N is the length of the source word and 2 is the size of the tagging set. In the conversion step, the number of possible candidates is $M^{N'}$, where N' is the number of chunks from the 1st step and M is the size of the tagging set. M is usually large, e.g., about 400 in Chinese and 50 in Japanese, and it is impossible to list all the candidates.

Our analysis shows that beyond the 10th candidate, almost all the probabilities of the candidates in both steps drop below 0.01. Therefore we decided to generate top-10 results for both steps to approximate the Equation 2.

3.3 Fast decoding algorithm

As introduced in the previous subsection, in the whole decoding process we have to perform n-best CRF decoding in the segmentation step and 10 n-best CRF decoding in the second CRF. Is it really necessary to perform the second CRF for all the segmentations? The answer is "No" for candidates with low probabilities. Here we propose a no-loss fast decoding algorithm for deciding when to stop performing the second CRF decoding.

Suppose we have a list of segmentation candidates which are generated by the 1st CRF, ranked by probabilities $P(A|S)$ in descending order $A : A_1, A_2, ..., A_N$ and we are performing the 2nd CRF decoding starting from A_1. Up to A_k, we get a list of candidates $T : T_1, T_2, ..., T_L$, ranked by probabilities in descending order. If we can guarantee that, even performing the 2nd CRF decoding for all the remaining segmentations $A_{k+1}, A_{k+2}, ..., A_N$, the top 1 candidate does not change, then we can stop decoding.

We can show that the following formula is the stop condition:

$$P_k(T_1|S) - P_k(T_2|S) > 1 - \sum_{j=1}^{k} P(A_j|S). \quad (3)$$

The meaning of this formula is that the probability of all the remaining candidates is smaller than the probability difference between the best and the second best candidates; on the other hand, even if all the remaining probabilities are added to the second best candidate, it still cannot overturn the top candidate. The mathematical proof is provided in Appendix A.

The stop condition here has no approximation nor pre-defined assumption, and it is a no-loss fast decoding algorithm.

4 Rapid development of a JSCM system

The JSCM represents how the source words and target names are generated simultaneously (Li et al., 2004):

$$\begin{aligned}& P(S, T) = P(s_1, s_2, ..., s_k, t_1, t_2, ..., t_k) \\ = \;& P(< s, t >_1, < s, t >_2, ..., < s, t >_k) \\ = \;& \prod_{k=1}^{K} P(< s, t >_k \mid < s, t >_1^{k-1})\end{aligned} \quad (4)$$

where $S = (s_1, s_2, ..., s_k)$ is a word in the source langauge and $T = (t_1, t_2, ..., t_k)$ is a word in the target language.

The training parallel data without alignment is first aligned by a Viterbi version EM algorithm (Li et al., 2004).

The decoding problem in JSCM can be written as:

$$\hat{T} = \arg\max_T P(S, T). \quad (5)$$

After the alignments are generated, we use the MITLM toolkit (Hsu and Glass, 2008) to build a trigram model with modified Kneser-Ney smoothing. We then convert the n-gram to a WFST M (Sproat et al., 2000; Caseiro et al., 2002). To allow transliteration from a sequence of characters, a second WFST T is constructed. The input word is converted to an acceptor I, and it is then combined with T and M according to $O = I \circ T \circ M$ where $\circ$ denotes the composition operator. The n–best paths are extracted by projecting the output, removing the epsilon labels and applying the n-shortest paths algorithm with determinization in the OpenFst Toolkit (Allauzen et al., 2007).

5 Experiments

We use several metrics from (Li et al., 2009) to measure the performance of our system.

1. Top-1 ACC: word accuracy of the top-1 candidate

2. Mean F-score: fuzziness in the top-1 candidate, how close the top-1 candidate is to the reference

3. MRR: mean reciprocal rank, 1/MRR tells approximately the average rank of the correct result

5.1 Comparison with the baseline and JSCM

We use the training, development and test sets of NEWS 2009 data for English-to-Chinese in our experiments as detailed in Table 1. This is a parallel corpus without alignment.

Training data	Development data	Test data
31961	2896	2896

Table 1: Corpus size (number of word pairs)

We compare the proposed decoding method with the baseline which uses only the best candidates in both CRF steps, and also with the well known JSCM. As we can see in Table 2, the proposed method improves the baseline top-1 ACC from 0.670 to 0.708, and it works as well as, or even better than the well known JSCM in all the three measurements.

Our experiments also show that the decoding time can be reduced significantly via using our fast decoding algorithm. As we have explained, without fast decoding, we need 11 CRF n-best decoding for each word; the number can be reduced to 3.53 (1 "the first CRF"+2.53 "the second CRF") via the fast decoding algorithm.

We should notice that the decoding time is significantly shorter than the training time. While testing takes minutes on a normal PC, the training of the CRF converter takes up to 13 hours on an 8-core (8*3G Hz) server.

Measure	Top-1 ACC	Mean F-score	MRR
Baseline	0.670	0.869	0.750
Joint optimization	**0.708**	**0.885**	**0.789**
JSCM	0.706	0.882	**0.789**

Table 2: Comparison of the proposed decoding method with the previous method and the JSCM

5.2 Further improvement

We tried to combine the two-step CRF model and the JSCM. From the two-step CRF model we get the conditional probability $P_{CRF}(T|S)$ and from the JSCM we get the joint probability $P(S,T)$. The conditional probability of $P_{JSCM}(T|S)$ can be calculuated as follows:

$$P_{JSCM}(T|S) = \frac{P(T,S)}{P(S)} = \frac{P(T,S)}{\sum_T P(T,S)}. \quad (6)$$

They are used in our combination method as:

$$P(T|S) = \lambda P_{CRF}(T|S) + (1-\lambda) P_{JSCM}(T|S) \quad (7)$$

where λ denotes the interpolation weight (λ is set by development data in this paper).

As we can see in Table 3, the linear combination of two sytems further improves the top-1 ACC to 0.720, and it has outperformed the best reported "Standard Run" (Li et al., 2009) result 0.717. (The reported best "Standard Run" result 0.731 used target language phoneme information, which requires a monolingual dictionary; as a result it is not a standard run.)

Measure	Top-1 ACC	Mean F-score	MRR
Baseline+JSCM	0.713	0.883	0.794
Joint optimization + JSCM	**0.720**	0.888	**0.797**
state-of-the-art (Li et al., 2009)	0.717	**0.890**	0.785

Table 3: Model combination results

6 Conclusions and future work

In this paper we have presented our new joint optimization method for a two-step CRF model and its fast decoding algorithm. The proposed

method improved the system significantly and outperformed the JSCM. Combining the proposed method with JSCM, the performance was further improved.

In future work we are planning to combine our system with multilingual systems. Also we want to make use of acoustic information in machine transliteration. We are currently investigating discriminative training as a method to further improve the JSCM. Another issue of our two-step CRF method is that the training complexity increases quadratically according to the size of the label set, and how to reduce the training time needs more research.

Appendix A. Proof of Equation 3

The CRF segmentation provides a list of segmentations: $A : A_1, A_2, ..., A_N$, with conditional probabilities $P(A_1|S), P(A_2|S), ..., P(A_N|S)$.

$$\sum_{j=1}^{N} P(A_j|S) = 1.$$

The CRF conversion, given a segmentation A_i, provides a list of transliteration output $T_1, T_2, ..., T_M$, with conditional probabilities $P(T_1|S, A_i), P(T_2|S, A_i), ..., P(T_M|S, A_i)$.

In our fast decoding algorithm, we start performing the CRF conversion from A_1, then A_2, and then A_3, etc. Up to A_k, we get a list of candidates $T : T_1, T_2, ..., T_L$, ranked by probabilities $P_k(T|S)$ in descending order. The probability $P_k(T_l|S)(l = 1, 2, ..., L)$ is accumulated probability of $P(T_l|S)$ over $A_1, A_2, ..., A_k$, calculated by:

$$P_k(T_l|S) = \sum_{j=1}^{k} P(A_j|S)P(T_l|S, A_j)$$

If we continue performing the CRF conversion to cover all N ($N \geq k$) segmentations, eventually we will get:

$$\begin{aligned} P(T_l|S) &= \sum_{j=1}^{N} P(A_j|S)P(T_l|S, A_j) \\ &\geq \sum_{j=1}^{k} P(A_j|S)P(T_l|S, A_j) \\ &= P_k(T_l|S) \end{aligned} \quad (8)$$

If Equation 3 holds, then for $\forall i \neq 1$,

$$\begin{aligned} P_k(T_1|S) &> P_k(T_2|S) + (1 - \sum_{j=1}^{k} P(A_j|S)) \\ &\geq P_k(T_i|S) + (1 - \sum_{j=1}^{k} P(A_j|S)) \\ &= P_k(T_i|S) + \sum_{j=k+1}^{N} P(A_j|S) \\ &\geq P_k(T_i|S) \\ &\quad + \sum_{j=k+1}^{N} P(A_j|S)P(T_i|S, A_j) \\ &= P(T_i|S) \end{aligned} \quad (9)$$

Therefore, $P(T_1|S) > P(T_i|S)(i \neq 1)$, and T_1 maximizes the probability $P(T|S)$.

References

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut and Mehryar Mohri 2007. *OpenFst: A General and Efficient Weighted Finite-State Transducer Library*. Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA), pages 11-23.

Diamantino Caseiro, Isabel Trancosoo, Luis Oliveira and Ceu Viana 2002. *Grapheme-to-phone using finite state transducers*. Proceedings IEEE Workshop on Speech Synthesis.

Asif Ekbal, Sudip Kumar Naskar and Sivaji Bandyopadhyay. 2006. *A modified joint source-channel model for transliteration*, Proceedings of the COLING/ACL, pages 191-198.

Bo-June Hsu and James Glass 2008. *Iterative Language Model Estimation: Efficient Data Structure & Algorithms*. Proceedings Interspeech, pages 841-844.

Kevin Knight and Jonathan Graehl. 1998. *Machine Transliteration*, Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando Pereira 2001. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.*, Proceedings of International Conference on Machine Learning, pages 282-289.

Haizhou Li, Min Zhang and Jian Su. 2004. *A joint source-channel model for machine transliteration*, Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics.

Haizhou Li, A. Kumaran, Vladimir Pervouchine and Min Zhang 2009. *Report of NEWS 2009 Machine Transliteration Shared Task*, Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009), pages 1-18

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

Richard Sproat 2000. *Corpus-Based Methods and Hand-Built Methods*. Proceedings of International Conference on Spoken Language Processing, pages 426-428.

Andrew J. Viterbi 1967. *Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm*. IEEE Transactions on Information Theory, Volume IT-13, pages 260-269.

Hanna Wallach 2002. *Efficient Training of Conditional Random Fields*. M. Thesis, University of Edinburgh.

Dong Yang, Paul Dixon, Yi-Cheng Pan, Tasuku Oonishi, Masanobu Nakamura and Sadaoki Furui 2009. *Combining a Two-step Conditional Random Field Model and a Joint Source Channel Model for Machine Transliteration*, Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009), pages 72-75