Transliteration by Sequence Labeling with Lattice Encodings and Reranking

Waleed Ammar Chris Dyer Noah A. Smith
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA, 15213, USA
{wammar,cdyer,nasmith}@cs.cmu.edu

Abstract

We consider the task of generating transliterated word forms. To allow for a wide range of interacting features, we use a conditional random field (CRF) sequence labeling model. We then present two innovations: a training objective that optimizes toward any of a set of possible correct labels (since more than one transliteration is often possible for a particular input), and a k-best reranking stage to incorporate nonlocal features. This paper presents results on the Arabic-English transliteration task of the NEWS 2012 workshop.

1 Introduction

Transliteration is the transformation of a piece of text from one language's writing system into another. Since the transformation is mostly explained as local substitutions, deletions, and insertions, we treat word transliteration as a **sequence labeling problem** (Ganesh et al., 2008; Reddy and Waxmonsky, 2009), using linear-chain conditional random fields as our model (Lafferty et al., 2001; Sha and Pereira, 2003). We tailor this model to the transliteration task in several ways.

First, for the Arabic-English task, each Arabic input is paired with multiple valid English transliteration outputs, any of which is judged to be correct. To effectively exploit these multiple references during learning, we use a training objective in which the model may favor some correct transliterations over the others. Computationally efficient inference is achieved by encoding the references in a lattice.

Second, inference for our first-order sequence labeling model requires a runtime that is quadratic in the number of labels. Since our labels are character n-grams in the target language, we must cope with thousands of labels. To make the most of each inference call during training, we apply a mini-batch training algorithm which converges quickly.

Finally, we wish to consider some global features that would render exact inference intractable. We therefore use a reranking model (Collins, 2000).

We demonstrate the performance benefits of these modifications on the Arabic-English transliteration task, using the open-source library `cdec` (Dyer et al., 2010)[1] for learning and prediction.

2 Problem Description

In the NEWS 2012 workshop, the task is to generate a list of ten transliterations in a specified target language for each named entity (in a known source language) in the test set. A training set is provided for each language pair. An entry in the training set comprises a named entity in the source language and one or more transliterations in the target language. Zhang et al. (2012) provides a detailed description of the shared task.

3 Approach

3.1 Character Alignment

In order to extract source-target character mappings, we use `m2m-aligner` (Jiampojamarn et al., 2007),[2] which implements a forward-backward algorithm to sum over probabilities of possible character sequence mappings, and uses Expectation Maximization to learn mapping probabilities. We allow source characters to be deleted, but not target characters. Parameters `-maxX` and `-maxY` are tuned on a devevelopment set.

Our running example is the Arabic name EAdl (in Buckwalter's ASCII-based encoding of Arabic) with two English transliterations: `ADEL` and `'ADIL`. The character alignment for the two pairs is shown in Fig. 1.

[1] `http://www.cdec-decoder.org`
[2] `http://code.google.com/p/m2m-aligner`

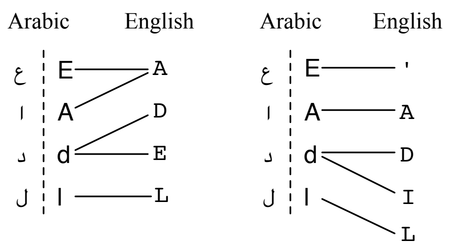

Figure 1: Character alignment for transliterating EAdl to ADEL and 'ADIL.

Type	Alignment	Labels
1:0	$x_i : \epsilon$	$y_i = \epsilon$
1:1	$x_i : t_j$	$y_i = t_j$
1:many	$x_i : t_j \dots t_k$	$y_i = t_j \dots t_k$
many:1	$x_i \dots x_p : t_j$	$y_p = t_j$ $y_i = \cdots = y_{p-1} = \delta$
many:many	$x_i \dots x_p : t_j \dots t_k$	$y_p = t_j \dots t_k$ $y_i = \cdots = y_{p-1} = \delta$

Table 1: Transforming alignments to sequence labels.

3.2 Sequence Labeling Scheme and Notation

We frame transliteration as a sequence labeling problem. However, transliteration is not a one-to-one process, meaning that a naïve application of one-label-per-token sequence models would be unlikely to perform well. Previous work has taken two different approaches. Reddy and Waxmonsky (2009) first segment the input character sequence, then use the segments to construct a transliteration in the target language. Since segmentation errors will compound to produce transliteration errors, we avoid this. Ganesh et al. (2008) do not require a segmentation step, but their model does not allow for many-to-one and many-to-many character mappings which are often necessary.

Our approach overcomes both these shortcomings: we have neither an explicit segmentation step, nor do we forbid many-to-many mappings. In our model, each character x_i in the source-language input $\mathbf{x} = \langle x_1, x_2, \dots, x_n \rangle$ is assigned a label y_i. However, a label y_i is a *sequence* of one or more target-language characters, a special marker indicating a deletion (ϵ), or a special marker indicating involvement in a many-to-one mapping (δ), that is, $y_i \in \Sigma^+ \cup \{\epsilon, \delta\}$, where Σ is the target language alphabet.[3] When an input $\mathbf{x}$ has multiple alternative reference transliterations, we denote the set $\mathcal{Y}^*(\mathbf{x}) = \{\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^K\}$.

We map the many-to-many alignments produced by `m2m-aligner` to one label for each input character, using the scheme in Table 1. Note that zero-to-one alignments are not allowed.

The two reference label sequences for our running example, which are constructed from the alignments in Fig. 1 are:

$\mathbf{x}$	$\mathbf{y}^1$	$\mathbf{y}^2$
E	δ	'
A	A	A
d	DE	DI
l	L	L

Of key importance in our model is defining, for each source character, the set of labels that can be considered for it. For each source character, we add all labels consistent with character alignments to the lexicon.

3.3 Model

Our model for mapping from inputs to outputs is a conditional random field (Lafferty et al., 2001), which defines the conditional probability of every possible sequence labeling $\mathbf{y}$ of a sequence $\mathbf{x}$ with the parametric form:

$$p_{\boldsymbol{\lambda}}(\mathbf{y} \mid \mathbf{x}) \propto \exp \textstyle\sum_{i=1}^{|\mathbf{x}|} \boldsymbol{\lambda} \cdot \mathbf{f}(\mathbf{x}, y_i, y_{i-1}) \qquad (1)$$

where $\mathbf{f}$ is a vector of real-valued feature functions.

3.4 Features

The feature functions used are instantiated by applying templates shown in Table 2 to each position i in the input string $\mathbf{x}$.

3.5 Parameter Learning

Given a training dataset of pairs $\{\langle \mathbf{x}_j, \mathbf{y}_j \rangle\}_{j=1}^{\ell}$ (note that each $\mathbf{y}$ is derived from the max-scoring character alignment), a CRF is trained to maximize the regularized conditional log-likelihood:

$$\max_{\boldsymbol{\lambda}} \mathcal{L}_{\{1,\dots,\ell\}}(\boldsymbol{\lambda}) \triangleq \textstyle\sum_{j=1}^{\ell} \log p_{\boldsymbol{\lambda}}(\mathbf{y}_j \mid \mathbf{x}_j) - C||\boldsymbol{\lambda}||_2^2 \qquad (2)$$

The regularization strength hyperparameter is tuned on development data. On account of the large data sizes and large label sets in several language pairs

[3]For an input type x, we only consider labels that were actually observed in the training data, which means the label set is finite.

Feature Template	Description
U1:y_i-x_i,	moving window of unigram, bigram and trigram context
U2:y_i-x_{i-1}-x_i,	
U3:y_i-x_i-x_{i+1},	
U4:y_i-x_{i-2}-x_{i-1}-x_i,	
U5:y_i-x_{i-1}-x_i-x_{i+1},	
U6:y_i-x_i-x_{i+1}-x_{i+2}	
U7:y_i, B1:y_i-y_{i-1}	label unigrams and bigrams
U8:$\|y_i\|$	label size (in characters)

Table 2: Feature templates for features extracted from transliteration hypotheses. The SMALLCAPS prefixes prevent accidental feature collisions.

(Table 3), batch optimization with L-BFGS is infeasible. Therefore, we use a variant of the mini-batch L-BFGS learning approach proposed by Le et al. (2011). This algorithm uses a series of randomly chosen mini-batches $\mathcal{B}^{(1)}, \mathcal{B}^{(2)}, \ldots$, each a subset of $\{1, \ldots, \ell\}$, to produce a series of weights $\boldsymbol{\lambda}^{(1)}, \boldsymbol{\lambda}^{(2)}, \ldots$ by running N iterations of L-BFGS on each mini-batch to compute the following:

$$\max_{\boldsymbol{\lambda}^{(i)}} \mathcal{L}_{\mathcal{B}^{(i)}}(\boldsymbol{\lambda}^{(i)}) - T\|\boldsymbol{\lambda}^{(i)} - \boldsymbol{\lambda}^{(i-1)}\|_2^2 \qquad (3)$$

The T parameter controls how far from the previous weights the optimizer can move in any particular mini-batch[4]. We use mini-batch sizes of 5, and start training with a small value of T and increase it as we process more iterations. This is equivalent to reducing the step-size with the number of iterations in conventional stochastic learning algorithms.

Language Pair	Unique Labels
Arabic-English	1,240
Chinese-English	2,985
Thai-English	1,771
English-Chinese	1,321
English-Japanese Kanji	4,572

Table 3: Size of the label set in some language pairs.

3.6 Using Multiple Reference Transliterations

In some language pairs, NEWS-2012 provides multiple reference transliterations in the training set. In this section, we discuss two possibilities for using these multiple references to train our transliteration model. The first possibility is to create multiple independent training inputs for each input $\mathbf{x}$, one for each correct transliteration in $\mathcal{Y}^*(\mathbf{x})$. Using this approach, with K different transliterations, the CRF training objective will attempt to assign probability $\frac{1}{K}$ to each correct transliteration, and 0 to all others (modulo regularization).

Alternatively, we can train the model to maximize the *marginal* probability assigned by the model to the *set* of correct labels $\mathcal{Y}^* = \{\mathbf{y}^1, \ldots, \mathbf{y}^K\}$. That is, we assume a set of training data $\{(\mathbf{x}_j, \mathcal{Y}_j^*)\}_{j=1}^{\ell}$ and replace the standard CRF objective with the following (Dyer, 2009):[5]

$$\max_{\boldsymbol{\lambda}} \sum_{j=1}^{\ell} \log \sum_{\mathbf{y} \in \mathcal{Y}_j^*} p_{\boldsymbol{\lambda}}(\mathbf{y} \mid \mathbf{x}_j) - C\|\boldsymbol{\lambda}\|_2^2 \quad (4)$$

This learning objective has more flexibility. It can maximize the likelihood of the training data by giving uniform probability to each reference transliteration for a given $\mathbf{x}$, but it does not have to. In effect, we do not care how probability mass is distributed among the correct labels. Our hope is that if some transliterations are difficult to model—perhaps because they are incorrect—the model will be able to disregard them.

To calculate the marginal probability for each $\mathbf{x}_j$, we represent $\mathcal{Y}^*(\mathbf{x})$ as a *label lattice*, which is supported as label reference format in `cdec`. A further computational advantage is that each $\mathbf{x}$ in the training data is now only a single training instance meaning that fewer forward-backward evaluations are necessary. The lattice encoding of both transliterations of our running example is shown in Fig. 2.

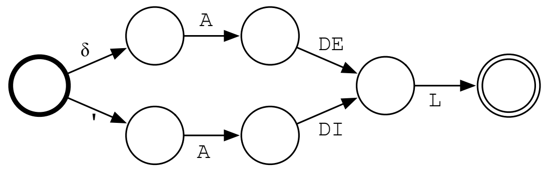

Figure 2: Lattice encoding two transliterations of EAdl: `ADEL` and `'ADIL`.

3.7 Reranking

CRFs require feature functions to be "local" to cliques in the underlying graphical model. One way to incorporate global features is to first decode the

[4]When $T = 0$, our learning algorithm is identical to the L-BFGS mini-batch algorithm of Le et al. (2011); however, we find that more rapid convergence is possible when $T > 0$.

[5]Unlike the standard CRF objective in eq. 2, the marginal probability objective is non-convex, meaning that we are only guaranteed to converge to a local optimum in training.

k-best transliterations using the CRF, then rerank based on global features combined with the CRF's conditional probability of each candidate. We experiment with three non-local features:

Character language model: an estimate of $p_{charLM}(\mathbf{y})$ according to a trigram character language model (LM). While a bigram LM can be factored into local features in a first order CRF, higher n-gram orders require a higher-order CRF.

Class language model: an estimate of $p_{classLM}(\mathbf{y})$, similar to the character LM, but collapses characters which have a similar phonetic function into one class (vowels, consonants, and hyphens/spaces). Due to the reduced number of types in this model, we can train a 5-gram LM.

Transliteration length: an estimate of $p_{len}(|\mathbf{y}| \mid |\mathbf{x}|)$ assuming a multinomial distribution with parameters estimated using transliteration pairs of the training set.

The probabilistic model for each of the global features is trained using training data provided for the shared task. The reranking score is a linear combination of $\log p_{crf}(\mathbf{y} \mid \mathbf{x})$, $\log p_{charLM}(\mathbf{y})$, $\log p_{classLM}(\mathbf{y})$ and $\log p_{len}(|\mathbf{y}| \mid |\mathbf{x}|)$. Linear coefficients are optimized using simulated annealing, optimizing accuracy of the 1-best transliteration in a development set. k-best lists are extracted from the CRF trellis using the lazy enumeration algorithm of Huang and Chiang (2005).

4 Experiments

We tested on the NEWS 2012 Arabic-English dataset. The train, development, and test sets consist of 27,177, 1,292, and 1,296 source named entities, respectively, with an average 9.6 references per name in each case.

Table 4 summarizes our results using the ACC score (Zhang et al., 2012) (i.e., word accuracy in top-1). "Basic CRF" is the model with mini-batch learning and represents multiple reference transliterations as independent training examples. We manually tuned the number of training examples and LBFGS iterations per mini-batch to five and eight, respectively. "CRF w/lattice" compactly represents the multiple references in a lattice, as detailed in §3.6. We consider reranking using each of the three global features along with the CRF, as well as the full set of four features.

Model	**Ar-En**
Basic CRF	23.5
CRF w/lattice	37.0
CRF w/lattice; rerank p_{crf}, p_{charLM}	40.7
CRF w/lattice; rerank $p_{crf}, p_{classLM}$	38.4
CRF w/lattice; rerank p_{crf}, p_{len}	37.3
CRF w/lattice, rerank all four	**42.8**

Table 4: Model performance, measured in word accuracy in top-1 (ACC, %).

Maximizing the marginal conditional likelihood of the set of alternative transliterations (rather than maximizing each alternative independently) shows a dramatic improvement in transliteration accuracy for Arabic-English. Moreover, in Arabic-English the basic CRF model converges in 120K mini-batch iterations, which is, approximately, seven times the number of iterations needed for convergence with lattice-encoded labels. A model converges when its ACC score on the development set ceases to improve in 800 mini-batch iterations. Results also show that reranking a k-best list of only five transliterations with any of the global features improves accuracy. Using all the features together to rerank the k-best list gives further improvements.

5 Conclusion

We built a CRF transliteration model that allows for many-to-many character mappings. We address limitations of CRFs using mini-batch learning and reranking techniques. We also show how to relax the learning objective when the training set contains multiple references, resulting in faster convergence and improved transliteration accuracy.

We suspect that including features of higher-order n-gram labels would help improve transliteration accuracy further, but it makes inference intractable due to the large set of labels. In future work, coarse transformations of label n-grams might address this problem.

Acknowledgments

This research was supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office under contract/grant number W911NF-10-1-0533. We thank anonymous reviewers for the valuable comments.

References

M. Collins. 2000. Discriminative reranking for natural language parsing. In *Proc. of ICML*.

C. Dyer, A. Lopez, J. Ganitkevitch, J. Weese, F. Ture, P. Blunsom, H. Setiawan, V. Eidelman, and P. Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proc. of ACL*.

C. Dyer. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *Proc. of NAACL*.

S. Ganesh, S. Harsha, P. Pingali, and V. Varma. 2008. Statistical transliteration for cross language information retrieval using HMM alignment and CRF. In *Proc. of the 2nd Workshop On Cross Lingual Information Access*.

L. Huang and D. Chiang. 2005. Better k-best parsing. In *In Proc. of the 9th International Workshop on Parsing Technologies*.

S. Jiampojamarn, G. Kondrak, and T. Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Proc. of NAACL*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng. 2011. On optimization methods for deep learning. In *Proc. of ICML*.

S. Reddy and S. Waxmonsky. 2009. Substring-based transliteration with conditional random fields. In *Proc. of the Named Entities Workshop*.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proc. of NAACL-HLT*.

M. Zhang, H. Li, M. Liu, and A. Kumaran. 2012. Whitepaper of NEWS 2012 shared task on machine transliteration.