Discriminative Methods for Transliteration

Dmitry Zelenko
SRA International
4300 Fair Lakes Ct.
Fairfax VA 22033
dmitry_zelenko@sra.com

Chinatsu Aone
SRA International
4300 Fair Lakes Ct.
Fairfax VA 22033
chinatsu_aone@sra.com

Abstract

We present two discriminative methods for name transliteration. The methods correspond to local and global modeling approaches in modeling structured output spaces. Both methods do not require alignment of names in different languages – their features are computed directly from the names themselves. We perform an experimental evaluation of the methods for name transliteration from three languages (Arabic, Korean, and Russian) into English, and compare the methods experimentally to a state-of-the-art joint probabilistic modeling approach. We find that the discriminative methods outperform probabilistic modeling, with the global discriminative modeling approach achieving the best performance in all languages.

1 Introduction

Name transliteration is an important task of transcribing a name from alphabet to another. For example, an Arabic "وليام", Korean "윌리엄", and Russian "Вильям" all correspond to English "William". We address the problem of transliteration in the general setting: it involves trying to recover original English names from their transcription in a foreign language, as well as finding an acceptable spelling of a foreign name in English.

We apply name transliteration in the context of cross-lingual information extraction. Name extractors are currently available in multiple languages. Our goal is to make the extracted names understandable to monolingual English speakers by transliterating the names into English.

The extraction context of the transliteration application imposes additional complexity constraints on the task. In particular, we aim for the transliteration speed to be comparable to that of extraction speed. Since most current extraction systems are fairly fast (>1 Gb of text per hour), the complexity requirement reduces the range of techniques applicable to the transliteration. More precisely, we cannot use WWW and the web count information to hone in on the right transliteration candidate. Instead, all relevant transliteration information has to be represented within a compact and self-contained transliteration model.

We present two methods for creating and applying transliteration models. In contrast to most previous transliteration approaches, our models are discriminative. Using an existing transliteration dictionary D (a set of name pairs $\{(\boldsymbol{f},\boldsymbol{e})\}$), we learn a function that directly maps a name $\boldsymbol{f}$ from one language into a name $\boldsymbol{e}$ in another language. We do not estimate either direct conditional $p(\boldsymbol{e}|\boldsymbol{f})$ or reverse conditional $p(\boldsymbol{f}|\boldsymbol{e})$ or joint $p(\boldsymbol{e},\boldsymbol{f})$ probability models. Furthermore, we do away with the notion of alignment: our transliteration model does not require and is not defined of in terms of aligned $\boldsymbol{e}$ and $\boldsymbol{f}$. Instead, all features used by the model are computed directly from the names $\boldsymbol{f}$ and $\boldsymbol{e}$ without any need for their alignment.

The two discriminative methods that we present correspond to local and global modeling paradigms for solving complex learning problems with structured output spaces. In the local setting, we learn linear classifiers that predict a letter e_i from the previously predicted letters $e_1 \ldots e_{i-1}$ and the original name $\boldsymbol{f}$. In the global setting, we learn a function W mapping a pair $(\boldsymbol{f},\boldsymbol{e})$ into a score $W(\boldsymbol{f},\boldsymbol{e}) \in R$. The function W is linear in features computed from the pair $(\boldsymbol{f},\boldsymbol{e})$. We describe the pertinent feature spaces as well as pre-

Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 612–617,
Sydney, July 2006.

sent both training and decoding algorithms for the local and global settings.

We perform an experimental evaluation for three language pairs (transliteration from Arabic, Korean, and Russian into English) comparing our methods to a joint probabilistic modeling approach to transliteration, which was shown to deliver superior performance. We show experimentally that both discriminative methods outperform the probabilistic approach, with global discriminative modeling achieving the best performance in all languages.

2 Preliminaries

Let E and F be two finite alphabets. We will use lowercase latin letters e, f to denote letters $e \in E$, $f \in F$, and we use bold letters $\mathbf{e} \in E^*$, $\mathbf{f} \in F^*$ to denote strings in the corresponding alphabets. The subscripted e_i, f_j denote ith and jth symbols of the strings $\mathbf{e}$ and $\mathbf{f}$, respectively. We use $\mathbf{e}[i,j]$ to represent a substring $e_i \ldots e_j$ of $\mathbf{e}$. If $j<i$, then $\mathbf{e}[i,j]$ is an empty string Λ.

A transliteration model is a function mapping a string $\mathbf{f}$ to a string $\mathbf{e}$. We seek to learn a transliteration model from a transliteration dictionary $D=\{(\mathbf{f},\mathbf{e})\}$. We apply the model in conjunction with a *decoding algorithm* that produces a string $\mathbf{e}$ from a string $\mathbf{f}$.

3 Local Transliteration Modeling

In local transliteration modeling, we represent a transliteration model as a sequence of local prediction problems. For each local prediction, we use the *history* h representing the context of making a single transliteration prediction. That is, we predict each letter e_i based on the pair $h=(\mathbf{e}[1,i-1], \mathbf{f}) \in H$.

Formally, we map $H \times E$ into a d-dimensional feature space $\varphi: H \times E \rightarrow R^d$, where each $\varphi_k(h,e) (k \in \{1,..,d\})$ corresponds to a condition defined in terms of the history h and the currently predicted letter e.

In order to model string termination, we augment E with a sentinel symbol \$, and we append \$ to each $\mathbf{e}$ from D.

Given a transliteration dictionary D, we transform the dictionary in a set of $|E|$ binary learning problems. Each learning problem L_e corresponds to predicting a letter $e \in E$. More precisely, for a pair $(\mathbf{f}[1,m],\mathbf{e}[1,n]) \in D$ and $i \in \{1,\ldots,n\}$, we generate a positive example $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e_i)$ for the learning problem L_e, where $e=e_i$, and a negative example $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e)$ for each L_e, where $e \neq e_i$.

Each of the learning problems is a binary classification problem and we can use our favorite binary classifier learning algorithm to induce a collection of binary classifiers $\{c_e : e \in E\}$. From most classifiers we can also obtain an estimate of conditional probability $p(e|h)$ of a letter e given a history h.

For decoding, in our experiments we use the beam search to find the sequence of letters (approximately) maximizing p(e|h).

3.1 Local Features

The features used in local transliteration modeling correspond to pairs of substrings of $\mathbf{e}$ and $\mathbf{f}$. We limit the length of substrings as well as their relative location with respect to each other.

- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e)$, generate a feature for every pair of substrings $(\mathbf{e}[i-w,i-1], \mathbf{f}[j-v,j])$, where $1 \leq w < W(E)$ and $0 \leq v < W(F)$ and $|i-j| \leq d(E,F)$. Here, $W(\cdot)$ is the upper bound on the length of strings in the corresponding alphabet, and $d(E,F)$ is the upper bound on the relative distance between substrings.
- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}[1,m]), e)$, generate the length difference feature $\varphi_{len}=i-m$. In experiments, we discretize φ_{len} to obtain 9 binary features: $\varphi_{len}=l$ $(l \in [-3,3])$, $\varphi_{len} \leq -4$, $4 \leq \varphi_{len}$.
- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}[1,m]), e)$, generate a language modeling feature $p(e| e_{[1,i-1]})$.
- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e)$ and $i=1$, generate "start" features: $(\hat{}f_1, \hat{}e)$, $(\hat{}f_1 f_2, \hat{}e)$.
- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e)$ and $i=2$, generate "start" features: $(\hat{}f_1, \hat{}e_1 e_2)$, $(\hat{}f_1 f_2, \hat{}e_1 e_2)$.
- For $\varphi((\mathbf{e}[1,i-1], \mathbf{f}), e)$ and $e=\$$, generate "end" features: $(f_m\$, e\$)$, $(f_{m-1} f_m\$, e\$)$.

The parameters $W(E)$, $W(F)$, and $d(E,F)$ are, in general, language-specific, and we will show, in the experiments, that different values of the parameters are appropriate for different languages.

4 Global Transliteration Modeling

In global transliteration modeling, we directly model the agreement function between **f** and **e**. We follow (Collins 2002) and consider the global feature representation $\Phi: F^* \times E^* \rightarrow R^d$.

Each global feature corresponds to a condition on the pair of strings. The value of a feature is the number of times the condition holds true for a given pair of strings. In particular, for every local feature $\varphi_k((\boldsymbol{e}[1,i-1], \boldsymbol{f}), e_i)$ we can define the corresponding global feature:

$$\Phi_k(\boldsymbol{f},\boldsymbol{e}) = \sum_i \varphi_k((\boldsymbol{e}[1,i-1], \boldsymbol{f}), e_i) \qquad (1)$$

We seek a transliteration model that is linear in the global features. Such a transliteration model is represented by d-dimensional weight vector $W \in R^d$. Given a string $\boldsymbol{f}$, model application corresponds to finding a string $\boldsymbol{e}$ such that

$$\boldsymbol{e} = \arg\max_{e'} \sum_k W_k \Phi_k(\boldsymbol{f},\boldsymbol{e}') \qquad (2)$$

As with the case of local modeling, due to computational constraints, we use beam search for decoding in global transliteration modeling.

(Collins 2002) showed how to use the Voted Perceptron algorithm for learning W, and we use it for learning the global transliteration model. We use beam search for decoding within the Voted Perceptron training as well.

4.1 Global Features

The global features used in local transliteration modeling directly correspond to local features described in Section 3.1.

- For $\boldsymbol{e}[1,n]$ and $\boldsymbol{f}[1,m]$, generate a feature for every pair of substrings $(\boldsymbol{e}[i-w,i],\boldsymbol{f}[j-v,j])$, where $1 \le w < W(E)$ and $0 \le v < W(F)$ and $|i-j| \le d(E,F)$.
- For $\boldsymbol{e}[1,n]$ and $\boldsymbol{f}[1,m]$, generate the length difference feature $\Phi_{len}=n-m$. In experiments, we discretize Φ_{len} to obtain 9 binary features: $\Phi_{len}=l$ $(l \in [-3,3])$, $\varphi_{len} \le -4$, $4 \le \varphi_{len}$.
- For $\boldsymbol{e}[1,n]$, generate a language modeling feature $(p(\boldsymbol{e}))^{1/n}$.
- For $\boldsymbol{e}[1,n]$ and $\boldsymbol{f}[1,m]$,, generate "start" features: $(\hat{}f_1,\hat{}e_1)$, $(\hat{}f_1f_2,\hat{}e_1)$, $(\hat{}f_1,\hat{}e_1e_2)$, $(\hat{}f_1f_2,\hat{}e_1e_2)$.
- For $\boldsymbol{e}[1,n]$ and $\boldsymbol{f}[1,m]$, generate "end" features: $(f_m\$,e_n\$)$, $(f_{m-1}f_m\$,e_n)$.

5 Joint Probabilistic Modeling

We compare the discriminative approaches to a joint probabilistic approach to transliteration introduced in recent years.

In the joint probabilistic modeling approach, we estimate a probability distribution $p(\boldsymbol{e},\boldsymbol{f})$. We also postulate hidden random variables $\boldsymbol{a}$ representing the alignment of $\boldsymbol{e}$ and $\boldsymbol{f}$. An alignment $\boldsymbol{a}$ of $\boldsymbol{e}$ and $\boldsymbol{f}$ is a sequence $a_1,a_2,\ldots a_L$, where $a_l = (\boldsymbol{e}[i_l-w_l,i_l],\boldsymbol{f}[j_l-v_l,j_l])$, $i_{l-1}+1=i_l-w_l$, and $j_{l-1}+1=j_l-v_l$. Note that we allow for at most one member of a pair a_l to be an empty string.

Given an alignment $\boldsymbol{a}$, we define the joint probability $p(\boldsymbol{e},\boldsymbol{f}|\boldsymbol{a})$:

$$p(\boldsymbol{e},\boldsymbol{f} \mid \boldsymbol{a}) = \prod_l p(\boldsymbol{e}[i_l - w_l, i_l], \boldsymbol{f}[j_l - v_l, j_l])$$

We learn the probabilities $p(\mathbf{e}[i_l-w_l,i_l],\mathbf{f}[j_l-v_l,j_l])$ using a version of EM algorithm. In our experiments, we use the Viterbi version of the EM algorithm: starting from random alignments of all string pairs in D, we use maximum likelihood estimates of the above probabilities, which are then employed to induce the most probable alignments in terms of the probability estimates. The process is repeated until the probability estimates converge.

During the decoding process, given a string $\boldsymbol{f}$, we seek both a string $\boldsymbol{e}$ and an alignment $\boldsymbol{a}$ such that $p(\boldsymbol{e},\boldsymbol{f}|\boldsymbol{a})$ is maximized. In our experiments, we used beam search for decoding.

Note that with joint probabilistic modeling use of a language model $p(\boldsymbol{e})$ is not strictly necessary. Yet we found out experimentally that an adaptive combination of the language model with the joint probabilistic model improves the transliteration performance. We thus combine the joint log-likelihood $\log(p(\boldsymbol{e},\boldsymbol{f}|\boldsymbol{a}))$ with $\log(p(\boldsymbol{e}))$:

$$score(e|f) = \log(p(\boldsymbol{e},\boldsymbol{f}|\boldsymbol{a})) + \alpha\log(p(\boldsymbol{e})) \qquad (3)$$

We estimate the parameter α on a held-out set by generating, for each $\boldsymbol{f}$, the set of top $K=10$ candidates with respect to $\log(p(\boldsymbol{e},\boldsymbol{f}|\boldsymbol{a}))$, then using (3) for re-ranking the candidates, and picking α to minimize the number of transliteration errors among re-ranked candidates.

6 Experiments

We present transliteration experiments for three language pairs. We consider transliteration from Arabic, Korean, and Russian into English. For all language pairs, we apply the same training and decoding algorithms.

6.1 Data

The training and testing transliteration dataset sizes are shown in Table 1. For Arabic and Russian, we created the dataset manually by keying in and translating Arabic, Russian, and English names. For Korean, we obtained a dataset of transliterated names from a Korean government website. The dataset contained mostly foreign

names transliterated into Korean. All datasets were randomly split into training and (blind) testing parts.

	Training	Testing
Arabic	935	233
Korean	11973	1363
Russian	545	121

Table 1. Transliteration Data.

Prior to transliteration, the Korean words of the Korean transliteration data were converted from their Hangul (syllabic) representation to Jamo (letter-based) representation to effectively reduce the alphabet size for Korean. The conversion process is completely automatic (see Unicode Standard 3.0 for details).

6.2 Algorithm Details

For language modeling, we used the list of 100,000 most frequent names downloaded from the US Census website. Our language model is a 5-gram model with interpolated Good-Turing smoothing (Gale and Sampson 1995).

We used the learning-to-classify version of Voted Perceptron for training local models (Freund and Schapire 1999). We used Platt's method for converting scores produced by learned linear classifiers into probabilities (Platt 1999). We ran both local and global Voted Perceptrons for 10 iterations during training.

6.3 Transliteration Results

Our discriminative transliteration models have a number of parameters reflecting the length of strings chosen in either language as well as the relative distance between strings. While we found that choice of $W(E)=W(F) = 2$ always produces the best results for all of our languages, the distance $d(E,F)$ may have different optimal values for different languages.

Table 2 presents the transliteration results for all languages for different values of d. Note that the joint probabilistic model does not depend on d. The results reflect the accuracy of transliteration, that is, the proportion of times when the top English candidate produced by a transliteration model agreed with the correct English transliteration. We note that such an exact comparison may be too inflexible, for many foreign names may have more than one legitimate English spelling. In future experiments, we plan to relax the requirement and consider alternative variants of transliteration scoring (e.g., edit distance, top-N candidate scoring).

	Local	Global	Prob
Arabic (d=1)	31.33	**32.61**	25.75
Arabic (d=2)	30.04	30.04	
Arabic (d=3)	26.61	27.03	
Korean (d=1)	26.93	30.44	26.93
Korean (d=2)	28.84	34.26	
Korean (d=3)	30.96	**35.28**	
Russian (d=1)	44.62	**46.28**	39.67
Russian (d=2)	38.84	41.32	
Russian (d=3)	38.01	38.01	

Table 2. Transliteration Results for Different Values of Relative Distance (d).

Table 2 shows that, for all three languages, the discriminative methods convincingly outperform the joint probabilistic approach. The global discriminative approach achieves the best performance in all languages. It is interesting that different values of relative distance are optimal for different languages. For example, in Korean, the Hangul-Jamo decomposition leads to fairly redundant strings of Korean characters thereby making transliterated characters to be relatively far from each other. Therefore, Korean requires a larger relative distance bound. In Arabic and Russian, on the other hand, transliterated characters are relatively close to each other, so the distance d of 1 suffices. While for Russian such a small distance is to be expected, we are surprised by such a small relative distance for Arabic. Our intuition was that omitting short vowels in spelling names in Arabic will increase d.

We have the following explanation of the low value of d for Arabic from the machine learning perspective: incrementing d implies adding a lot of extraneous features to examples, that is, increasing attribute noise. Increased attribute noise requires a corresponding increase in the number of training examples to achieve adequate performance. While for Korean the number of training examples is sufficient to cope with the attribute noise, the relatively small Arabic training sample is not. We hypothesize that with increasing the number of training examples for Arabic, the optimal value of d will also increase.

7 Related Work

Most work on name transliteration adopted a source-channel approach (Knight and Grael 1998; Al-Onaizan and Knight 2002a; Virga and Khudanpur 2003; Oh and Choi 2000) incorporat-

ing phonetics as an intermediate representation. (Al-Onaizan and Knight 2002) showed that use of outside linguistic resources such as WWW counts of transliteration candidates can greatly boost transliteration accuracy. (Li *et al*. 2004) introduced the joint transliteration model whose variant augmented with adaptive re-ranking we used in our experiments.

Among direct (non-source-channel) models, we note the work of (Gao et al. 2004) on applying Maximum Entropy to English-Chinese transliteration, and the English-Korean transliteration model of (Kang and Choi 2000) based on decision trees.

All of the above models require alignment between names. We follow the recent work of (Klementiev and Roth 2006) who addressed the problem of discovery of transliterated named entities from comparable corpora and suggested that alignment may not be necessary for transliteration.

Finally, our modeling approaches follow the recent work on both local classifier-based modeling of complex learning problems (McCallum et al. 2000; Punyakanok and Roth 2001), as well as global discriminative approaches based on CRFs (Lafferty et al. 2001), SVM (Taskar et al. 2005), and the Perceptron algorithm (Collins 2002) that we used in our experiments.

8 Conclusions

We presented two novel discriminative approaches to name transliteration that do not employ the notion of alignment. We showed experimentally that the approaches lead to superior experimental results in all languages, with the global discriminative modeling approach achieving the best performance.

The results are somewhat surprising, for the notion of alignment seems very intuitive and useful for transliteration. We will investigate whether similar alignment-free methodology can be extended to full-text translation. It will also be interesting to study the relationship between our discriminative alignment-free methods and recently proposed discriminative alignment-based methods for transliteration and translation (Taskar et al. 2005a; Moore 2005).

We also showed that for name transliteration, global discriminative modeling is superior to local classifier-based discriminative modeling. This may have resulted from poor calibration of scores and probabilities produced by individual classifiers. We plan to further investigate the relationship between the local and global approaches to complex learning problems in natural language.

References

Y. Al-Onaizan and K. Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. *Proceedings of ACL.*

Y. Al-Onaizan and K. Knight. 2002a. Machine Transliteration of Names in Arabic Text. *Proceedings of ACL Workshop on Computational Approaches to Semitic Languages.*

M. Collins. 2002. Discriminative Training for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of EMNLP.*

Y. Freund and R. Shapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37, 277–296.

W. Gale and G. Sampson. 1995. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics* 2:217-235.

Gao Wei, Kam-Fai Wong, and Wai Lam. 2004. Phoneme-based transliteration of foreign names for OOV problem. *Proceedings of the First International Joint Conference on Natural Language Processing*.

B.J. Kang and Key-Sun Choi, 2000. Automatic Transliteration and Back-transliteration by Decision Tree Learning, *Proceedings of the 2nd International Conference on Language Resources and Evaluation*.

A. Klementiev and D. Roth. 2006. Named Entity Transliteration and Discovery from Multilingual Comparable Corpora. *Proceedings of ACL.*

K. Knight and J. Graehl. 1998. Machine Transliteration, *Computational Linguistics*, 24(4).

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of the Eighteenth International Conference on Machine Learning.*

Li Haizhou, Zhang Min, and Su Jian. 2004. A Joint Source-channel Model for Machine Transliteration. *Proceedings of ACL 2004.*

A. McCallum, D. Freitag, and F. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. *Proceedings of ICML.*

R. Moore. 2005. A Discriminative Framework for Bilingual Word Alignment. *Proceedings of the Conference on Empirical Methods in Natural Language Processing.*

Jong-Hoon Oh and Key-Sun Choi. 2000. An English-Korean Transliteration Model Using Pronunciation and Contextual Rules. *Proceedings of COLING*.

J. Platt. 1999. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In *Advances in Large Margin Classi£ers*.

V. Punyakanok and D. Roth. 2001. The Use of Classifiers in Sequential Inference. *Proceedings of the Conference on Advances in Neural Information Processing Systems*.

B. Taskar, V. Chatalbashev, D. Koller and C. Guestrin. 2005. Learning Structured Prediction Models: A Large Margin Approach. *Proceedings of Twenty Second International Conference on Machine Learning*.

B. Taskar, S. Lacoste-Julien, and D. Klein. 2005a. A Discriminative Matching Approach to Word Alignment. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

P. Virga and S. Khudanpur. 2003. Transliteration of Proper Names in Cross-lingual Information Retrieval. *Proceedings of ACL 2003 workshop MLNER*.