A Dirichlet Process Mixture Based Name Origin Clustering and Alignment Model for Transliteration

Chunyue Zhang, Tiejun Zhao, and Tingting Li

School of Computer Science and Technology, Harbin Institute of Technology, No. 92, West Dazhi St., Harbin 150001, China

Correspondence should be addressed to Tiejun Zhao; tjzhao@hit.edu.cn

Received 25 November 2014; Revised 28 April 2015; Accepted 16 June 2015

Academic Editor: Panayiotis G. Georgiou

Copyright © 2015 Chunyue Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In machine transliteration, it is common that the transliterated names in the target language come from multiple language origins. A conventional maximum likelihood based single model can not deal with this issue very well and often suffers from overfitting. In this paper, we exploit a coupled Dirichlet process mixture model (cDPMM) to address overfitting and names multiorigin cluster issues simultaneously in the transliteration sequence alignment step over the name pairs. After the alignment step, the cDPMM clusters name pairs into many groups according to their origin information automatically. In the decoding step, in order to use the learned origin information sufficiently, we use a cluster combination method (CCM) to build clustering-specific transliteration models by combining small clusters into large ones based on the perplexities of name language and transliteration model, which makes sure each origin cluster has enough data for training a transliteration model. On the three different Western-Chinese multiorigin names corpora, the cDPMM outperforms two state-of-the-art baseline models in terms of both the top-1 accuracy and mean F-score, and furthermore the CCM significantly improves the cDPMM.

1. Introduction

Machine transliteration concentrates on translating a word or phrase from one writing system to another based on the pronunciation, which is a major way of importing foreign words such as proper nouns and technical terms into the target languages. It is also a major method for out-of-vocabulary words translation which commonly consists of people names, places names, company names, and products names. Machine transliteration is an essential task for many NLP applications such as automated question answering (QA), machine translation (MT), cross language information retrieval (CLIR), and information extraction (IE).

Machine transliteration has emerged around many years ago as a part of machine translation. The methods used for transliteration were often based on the phonetic origin of source and target languages at first, but now the spelling-based statistical approaches have nearly dominated this field. Because the spelling-based methods directly align characters on the training corpus based on the statistical results and do not require language specific phonetic knowledge, they are language independent and capable of achieving state-of-the-art performance [1]. However, there are still several common challenges [2] in spelling-based machine transliteration, such as script specifications, language of origin, missing sounds, and overfitting. In this paper, we concentrate on the language of origin and overfitting problems.

The origin problem of the language, also known as multiorigin identification, can affect both the alignment step and decoding step of statistical transliteration. In the Western-Chinese machine transliteration, the bilingual training corpus usually contains name pairs originated from more than one language. Names coming from different languages have their own transliteration pronunciation rules. For example,

“kim jong-il/金正恩” (Korea),
“kana Gaski/金崎” (Japan),
“haw king/霍金” (English),
“jin yong/金庸” (China),
“mundinger/蒙丁格” (English),
“ding guo/丁果” (Chinese).
The same Chinese character “金” should be aligned to different romanized character sequences: “金: kim,” “金: kana,” “金: king,” and “金: jin.” In this case, it is not easy for a transliteration model, which does not take the names origin into consideration, to learn the right alignment for “金” and it is the same difficulty for the character “丁.”

Overfitting is another problem which exists in many previous alignment models such as GIZA++ [3], M2M-aligner [4], and HMM [5] in transliteration. These alignment models are optimized based on expectation maximization and try to fit the training data by maximum likelihood (ML) estimation. In practice the training data usually has some kind of deficiency, such as containing noise data and being not large enough. So the training data often cannot hold the real data distribution. When a ML model learns on this kind of training dataset, it often suffers from overfitting.

In this paper, we exploit a simple and fully unsupervised model to solve the two problems above, which is able to both cluster and align simultaneously. The coupled Dirichlet process mixture model (cDPMM) [6] integrates a Dirichlet mixture model for name pairs clustering and a set of Bayesian bilingual alignment models (BBAM) [7] for each cluster. The clustering and alignment models synergistically complement one another: the clustering model groups the training data into the right origin cluster so that self-consistent alignment models can be built on these data of the same type, and at the same time the alignment probabilities from the alignment models can direct the clustering process. Furthermore, based on the cluster and alignment results, we propose a cluster combination method (CCM) to build the cluster-specific transliteration model and language model for each cluster. In the decoding step, given a source name, we classify it into the most similar cluster based on the language model and transcribe it with the cluster-specific transliteration model. In this way, we use the origin information to direct the alignment and decoding step and obtain significant improvements.

Our major contributions in this paper are summarized as follows:

(i) We exploit a Dirichlet process mixture model for clustering name pairs, which is fully unsupervised and does not require setting the cluster numbers beforehand. It is effectively capable of discovering an appropriate number of clusters from the data automatically.

(ii) We apply the Bayesian segmentation alignment model in the alignment step over each cluster, which allows many-to-many monotonic alignment and can overcome overfitting.

(iii) The clustering and alignment models are coupled in a unified model, and they work synergistically to support each other.

(iv) By combining the small clusters into large ones based on the perplexities of name language models and transliteration models, we build the combined cluster-specific transliteration model and language model and can transliterate each source name based on its origin.

(v) We conduct experiments on the three Western to Chinese multiorigin datasets and the results show that our cDPMM and CCM methods are competitive in terms of accuracy and mean $F$-score compared to other methods.

The rest of the paper is organized as follows: we start by introducing related works in Section 2. Then we describe the detail of our coupled Dirichlet process mixture model in Section 3. In Section 4 we introduce the cluster combined method which can obtain large size of training data for building cluster-specific transliteration model and language model. Section 5 provides the experimental setup, results, and analysis. Finally, we conclude this paper and discuss the future work.

2. Related Work

Machine transliteration methods can be categorized into phonetic-based models [8], spelling-based models [9], and hybrid models which utilize both the phonetic and spelling information [10, 11]. Among them, spelling-based models, which directly align characters in the training corpus based on the statistical learning, have been a popular method in transliteration because it is language independent and phonetic knowledge independent, and the performance often is significantly higher compared with phonetic-based models.

In some cases, transliteration can be viewed as a special instance of machine translation, so the aligning and decoding methods in statistical machine translation (SMT) [12] can be used in transliteration. Many previous works [13, 14] have built transliteration models based on GIZA++ [3] and Moses decoder [12]. In these works, a single character or syllable in a name pair is treated as a word in a sentence pair, and a monotonic many-to-many alignment is carried out. As the alignment is monotonic and the decoding is simpler in machine transliteration, many transliteration-specific methods have been proposed, such as Alpha-Beta Model [15], Joint Source Channel Model [16], and M2M-aligner [4]. These models are optimized based on expectation maximization and always have the overfitting problem.

Recently, the nonparametric Bayesian model is widely used in many natural language process (NLP) tasks and achieves competitive results. Someone has tried to use this kind of model to address the transliteration. A Bayesian bilingual alignment model (BBAM) [7] is proposed to segment and align the training name pair, which uses Dirichlet process to model the alignment sequence and treats basic transliteration units as samples, and applies a blocked version of a Gibbs sampler [17] to get each transliteration unit. The BBAM has been integrated to many transliteration methods to improve transliteration task [18] and transliteration mining task [19]. In [20], Huang et al. propose a nonparametric Bayesian method to train asynchronous adaptor grammars for transliteration. In this paper, we also integrate the BBAM [7] to obtain transliteration alignment for easing the overfitting issues.

The multiorigin problem in transliteration is firstly proposed by Huang et al. [20]. They choose an unsupervised
bottom-up hierarchical clustering algorithm and use the language and translation model to calculate the similarity of every two clusters, and several cluster-specific translation and classification models are built based on the clustering results. Li et al. [21] propose a supervised classification model to classify the name pairs based on their language origins and genders, which chooses the most similar cluster-specific model for a source name. Hagiwara and Sekine propose a latent semantic transliteration model (DM-LST) [22] to integrate the clustering and alignment model, which is an extension to their latent class transliteration model [23] and applies Dirichlet mixture as a prior distribution for distribution of alignment units to address overfitting.

3. Coupled Dirichlet Process Mixture Model

In [6], Li et al. propose a new transliteration model called Coupled Dirichlet process mixture model (cDPMM), which simultaneously clusters and bilingually aligns training data. And it is a fully unsupervised method which can discover an appropriate number of clusters for the training data. In this section we briefly make a description of this model. We first give the terminologies that will be used in the following section. Then we describe the detail of cDPMM; in Section 3.2 the Bayesian bilingual alignment model (BBAM) is presented, and in Section 3.3 we describe the cDPMM for clustering and the strategy to couple the BBAM.

3.1. Terminology. In this work, we concentrate on solving the overfitting and multiorigin problems in transliteration. Firstly, we denote the training set itself as a set of sequence-pairs: \( D = \{ x_{ij} \}_{i=1} \), where \( I \) is the name pairs size of training data. The cDPMM will cluster and segment every bilingual name pair into bilingual character sequence-pair in the transliteration alignment stage. As in [6], we call every character sequence-pair Transliteration Unit (TU). We denote the source and target side of a TU as \( s_{ij} = (s_{1},...,s_{m}) \) and \( t_{ij} = (t_{1},...,t_{n}) \), respectively, where \( s(t) \) is a single character in source (target) name. So a TU can be denoted as \( tu = (s,t) = ((s_{1},...,s_{m}),(t_{1},...,t_{n})) \). We use the same notation to denote a transliteration pair \( x = tu^{L} \), where \( L \) is the number of TUs used to segment \( x \). Our aim is to obtain a bilingual transliteration pair \( x = tu^{L} \), where \( L \) is the number of TUs used to segment \( x \) and \( x_{ij} \) is a single character in source (target) name. We use \( y = ((s_{1},t_{1}),...,(s_{L},t_{L})) \) to indicate one derivation for \( x \), and for each \( x_{ij} \) there is a derivation set \( R_{x_{ij}} = \{ y_{1},y_{2},...,y_{n} \} \). For each cluster obtained by cDPMM, we use \( k \) to indicate the cluster ID. \( k \) to indicate the set of name pairs in this cluster, \( \theta_{k} \) to represent the alignment model parameters of cluster \( k \), and \( D_{k} = \{ x_{ij} \}_{i=1} \) to represent the clustered name pair dataset.

3.2. Alignment Model. In this work, we use a Bayesian model [7] in the alignment step, which implements bilingual segmentation and alignment based on forward filter and backward sampling algorithm to get TUs, and use the multi-}

3.2.1. Multinomial Dirichlet Process. The alignment component of our cDPMM is a multinomial Dirichlet process. In our work, the Dirichlet process is a stochastic process defined over the set \( T \) of all possible TUs in the training dataset and its sample path is a probability distribution on the set \( T \). Our model has two basic components: one model for generating a TU which has been generated at least once and the other one for assigning a probability to a TU that has not yet been seen. The probability of generating a new TU should be considerably lower than that of generating the other old TUs, and the more TUs are generated the more reliable and complete the prior distribution is. So the model prefers the TUs which have been seen more than the unseen ones. Theoretically, our model can generate any sequence-pair of the training dataset in the bilingual language. The Dirichlet process segmentation and alignment model can be written in the following form:

\[
G | \alpha, G_0 \sim DP(\alpha, G_0),
\]

\[
(s, t) | G \sim G.
\]
We use (3) to generate the $j$th TU $(\tilde{s}_j, \tilde{t}_j)$. It gives a probability to $(\tilde{s}_j, \tilde{t}_j)$ based on the property of Dirichlet process and the other TUs seen in the history so far:

$$p((\tilde{s}_j, \tilde{t}_j) | (\tilde{s}_{-j}, \tilde{t}_{-j})) = \frac{N((\tilde{s}_j, \tilde{t}_j)) + \alpha_{G_0}(\tilde{t}_j)}{N + \alpha}, \tag{3}$$

where $N$ is the total number of TUs generated so far and $N((\tilde{s}_j, \tilde{t}_j))$ is the count number of $(\tilde{s}_j, \tilde{t}_j)$ which has been seen in the history. $(\tilde{s}_{-j}, \tilde{t}_{-j})$ are all the TUs generated so far except $(\tilde{s}_j, \tilde{t}_j)$. $G_0$ and $\alpha$ are the base measure and concentration parameter as before.

3.2.3. The Blocked Gibbs Sampling. In the BBAM the basic samples are TUs. However the training data in our translation problem are name pairs, and a Blocked Gibbs Sampling [25] Algorithm is used here to obtain a whole TU sequence for a name pair. Algorithm 1 is our Blocked Gibbs Sampling Algorithm. We use the last updated alignment model, which does not include the current $x$ to be sampled, to calculate the probability for each derivation $y = ((\tilde{s}_1, \tilde{t}_1), (\tilde{s}_2, \tilde{t}_2), \ldots, (\tilde{s}_L, \tilde{t}_L))$ of $x$ based on

$$p(y) = \prod_{j=1}^{L} p((\tilde{s}_j, \tilde{t}_j)). \tag{4}$$

Finally, a certain $y'$ is chosen based on the derivation probability distribution of $x$. In this paper, a forward filtering and backward sampling (FFBS) dynamic program [25] with $n$-grams information is used to obtain a bilingual segmentation path.

Figure 1 is an example of all possible segmentation paths of the name pair $x$ “Elsa-爱丽莎.” Each node represents a subsequence $((\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_j), (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_j))$ of $x$, which records the probability of all the segmentations for this subsequence, and we use forward$[m][n]$ to denote it and set forward$[0][0] = 1$. Equation (5) is used to calculate forward

$$\text{Forward} [m][n] = \sum \text{forward} [m-|\tilde{s}_j|][n-|\tilde{t}_j|] p((\tilde{s}_j, \tilde{t}_j)). \tag{5}$$

Each arc represents a bilingual TU $(\tilde{s}_j, \tilde{t}_j)$, and the figure on the arc represents log-probability log $p(\tilde{s}_j, \tilde{t}_j)$ of the alignment. The path from the head nodes to the end nodes represents a whole derivation $y$ of $x$.

The backward sampling step is to choose a sampling path over all possible paths from Figure 1. The algorithm samples the path from the tail to the head nodes (to obtain the $y$ from $(\tilde{s}_L, \tilde{t}_L)$ to $(\tilde{s}_1, \tilde{t}_1)$). If we have obtained $L-j$ TUs $((\tilde{s}_{j+1}, \tilde{t}_{j+1}), \ldots, (\tilde{s}_L, \tilde{t}_L))$, for the $j$th TU $(\tilde{s}_j, \tilde{t}_j)$, we use (6) to calculate the probability of candidate arc of $(\tilde{s}_j, \tilde{t}_j)$:

$$\text{backward} [m-[\tilde{s}_j]][n-[\tilde{t}_j]] = \text{forward} [m-|\tilde{s}_j|][n-|\tilde{t}_j|] p((\tilde{s}_j, \tilde{t}_j)). \tag{6}$$

Then we sample a certain $(\tilde{s}_j, \tilde{t}_j)$ based on the probability distribution.

3.3. The Clustering Model. The clustering component of our cDPMM is a Dirichlet process mixture model (DPMM). In our case, the DPMM is a stochastic process defined over a set of bilingual name pairs $D$ and a sample path is a probability distribution of name pair origin on $D$. Furthermore, the definition of Dirichlet process mixture model is shown as follows:

$$G | \alpha, G_0 \sim \text{DP}(\alpha, G_0), \tag{7}$$

$$\theta | G \sim G,$$

$$x_i | \theta_k \sim F(x_i | \theta_k).$$

The DPMM model computes the distribution of a set of observation name pairs $D = x_1, x_2, \ldots, x_n$ by using a set of latent parameters $\theta = \theta_1, \theta_2, \ldots, \theta_K$. And each $\theta_k$ is drawn independently and identically from $G$; each $\theta_k$ is drawn independently from a distribution $F(\theta_k)$ parametrized by $\theta_k$. As the Dirichlet process, the DPMM also has two basic components: one model for generating an $x_i$ from clusters which have been generated and one for assigning a probability to $x_i$ for being generated by a new cluster.

**Algorithm 1:** The Blocked Gibbs Sampling Algorithm.

Input: Random initial corpus segmentation

Output: Unsupervised co-segmentation of the corpus according to the model

1. for each iter $\in [1, \text{NumIterations}]$ do
2. for each bilingual word-pair $x \in D$ do
3. for co-segmentation of $y_i$ of $x$ do
4. Compute probability $p(y_i | h)$;
5. Where $h$ is the distribution of all TUs that have been generated before except that from $x$;
6. end for
7. Sample a $y_i$ from the distribution of $p(y_i | h)$;
8. Update counts;
9. end for
10. end for

We use (3) to generate the $j$th TU $(\tilde{s}_j, \tilde{t}_j)$. It gives a probability to $(\tilde{s}_j, \tilde{t}_j)$ based on the property of Dirichlet process and the other TUs seen in the history so far:

$$p((\tilde{s}_j, \tilde{t}_j) | (\tilde{s}_{-j}, \tilde{t}_{-j})) = \frac{N((\tilde{s}_j, \tilde{t}_j)) + \alpha_{G_0}(\tilde{t}_j)}{N + \alpha}, \tag{3}$$

where $N$ is the total number of TUs generated so far and $N((\tilde{s}_j, \tilde{t}_j))$ is the count number of $(\tilde{s}_j, \tilde{t}_j)$ which has been seen in the history. $(\tilde{s}_{-j}, \tilde{t}_{-j})$ are all the TUs generated so far except $(\tilde{s}_j, \tilde{t}_j)$. $G_0$ and $\alpha$ are the base measure and concentration parameter as before.
The probability of a cluster \( k \) which generates a name pair \( x_i \) is decided by the similarity between \( x_i \) and the name pairs which have been assigned to cluster \( k \).

As our DPMM, the CRP can also be used here. Every name pair \( x_i \) corresponds to a customer, and the dish served on each table corresponds to an origin of name pairs. We use \( z = (z_1, z_2, \ldots, z_I) \), \( z_i \in 1, \ldots, K \), to indicate the cluster ID for each transliteration pair \( x_i \) in training set which have been assigned some cluster and \( \theta = \theta_1, \theta_2, \ldots, \theta_K \), to represent the parameters of the component associated with each cluster. The generating probability of a transliteration pair \( x_i \) for cluster \( k \) is calculated as follows:

\[
p(z_i = k \mid D, \theta, z_{-i}) \propto \frac{n_k}{n - 1 + \alpha} F(x_i \mid \theta_k),
\]

\[
p(z_i = K + 1 \mid D, \theta, z_{-i}) \propto \frac{1}{n - 1 + \alpha} F(x_i \mid \theta_{K+1}),
\]

where \( n_k \) is the number of transliteration pairs in the existing cluster \( k \in 1, \ldots, K \); \( K \) is the number of clusters that are obtained by now. \( z_i \) is the cluster indicator for \( x_i \), and \( z_{-i} \) is the sequence of observed clusters up to \( x_i \).

In our cDPMM, each mixture component is a multinomial DP model, and we use the alignment method introduced in Section 3.2 to train the alignment model for each cluster. For a component \( k \), it has a distribution of TUs and the parametric \( \theta_k \) is unfixed and changes as the transliteration pairs that newly enter into the cluster. For a new component we use (2) to calculate the probability of each TU. We decide the probability of a name pair \( x_i \) belongs to a cluster \( k \) based on the joint channel model. There is a set of TUs derivation paths \( R_x = \{ y \mid y = ((s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{il}, t_{il})) \} \) where \( x_i = (s_{i1} s_{i2} \ldots s_{il} t_{i1} t_{i2} \ldots t_{il}) \) for each name pair \( x_i \), and the similarity between \( x_i \) and a cluster is calculated as follows:

\[
f(x_i \mid \theta_k) = \sum_{y \in R_x} \prod_{(s,t) \in y} p((s, t) \mid \theta_k),
\]

\[
f(x_i \mid \theta_{K+1}) = \sum_{y \in R_x} \prod_{(s,t) \in y} G_0((s, t) \mid \theta_{K+1}).
\]

The input of our cDPMM is a dataset of name pairs \( x_i = (s_{i1}, t_{i1}) \); the output is the clustering result of all \( x_i \) and the alignment model in each cluster. In the sampling step, the cDPMM will first sample a cluster \( k \) based on the cluster distribution decided by (8), and a derivation \( y \) will be sampled for \( x_i \) based on probability calculated by (9).

4. Cluster Combination Method and Source Name Classification Model

In [6], Li et al. just use the cDPMM to direct the alignment step. In order to make full use of clustering and alignment results of cDPMM, in this paper we train a transliteration model for each name origin cluster. However, based on the theory of unsupervised clustering model, we will get a lot of clusters with small number of name pairs. In order that each cluster has enough name pairs to train a reliable model, we propose a method that combines the clusters generated by the cDPMM and force the size of each origin cluster at least one-tenth of the whole training set when decoding a source name. Also in order to classify the source name to the suitable origin...
cluster, the origin-specific language model is trained on each cluster and used for classification.

4.1. Cluster Combination Model (CCM). We use the language model (LM) and transliteration model (TM) perplexities, which is used by Huang et al. [20] between two clusters to decide the similarity of them. We use $D_k = \{ (S_{k1}^1, t_{k1}^1), \ldots, (S_{kn}^k, t_{nk}^k) \}$ to denote the data set in cluster $k$; $n_k$ is the size of dataset $D_k$. For a cluster $k$, we can train two language models of source and target names and have an alignment model. We use $\theta_k = \{ \lambda_{sk}, \lambda_{tk}, \lambda_{ek} \}$ to indicate the model parameters; here $\lambda_{sk}(p_{sk})$ is the language model trained by the source and target names, and $\lambda_{ak}$ is the alignment model based on our cDPMM. The formulas of the perplexities definition between clusters $k_1$ and $k_2$ are shown as follows:

$$d(k_1, k_2) = -\frac{1}{D_{k_1}} \log P(D_{k_1} | \theta_{k_1}) - \frac{1}{D_{k_2}} \log P(D_{k_2} | \theta_{k_2}),$$

(10)

$$P(D_{k_1}, \theta_{k_1}) \propto \sum_{i=1}^{n_{k_1}} \log \left[ \lambda_{sk}(S_{ki}^1) \lambda_{tk}(t_{ki}^1 | S_{ki}^1) \right] + \lambda_{ak}(\hat{t}_{ki}^1) \lambda_{sk}(\hat{S}_{ki}^1 | S_{ki}^1).$$

(11)

Here $d(k_1, k_2)$ is the distance function between clusters $k_1$ and $k_2$. We greedily combine two clusters into a large one by distance $d$ until the size of each cluster reaches one-tenth of the whole training dataset.

4.2. Source Name Classification Model. For a source name $\tilde{s} = \{s_1, s_2, \ldots, s_m\}$ we will choose the most similar cluster for it. We use the language model trained by the source names in each cluster to classify the source names and choose the transliteration model with the highest language model probability. The calculation equation is shown as follows:

$$\hat{t}^i = \arg \max_{i \in K} \left\{ P(\tilde{s} | \theta_i) P(\theta_i) = \arg \max_{i \in K} p_{si}(\tilde{s}) I_i, \right\}$$

(12)

where the $P(\theta_i)$ is the prior distribution of the cluster, $I_i$ is the number of name pairs of cluster $i$, and $I$ is the number of the whole datasets.

5. Experiments

In this section, we conduct a set of experiments to validate the performance of our cDPMM and CCM. First, we choose several datasets with different name origins information to make sure our experiments results are convincing. Second, we analyze the cluster and alignment results of cDPMM. Third, we utilize the results of cDPMM in several ways to train two state-of-the-art decoders to transliterate the test sets. After comparing cDPMM with GIZA++ and BBAM, we conduct the experiments in the same dataset to test our CCM.

5.1. Data and Model Setup

5.1.1. Corpora. To empirically validate our approach, we investigate the result of our model by conducting Western-Chinese name transliteration on three corpora containing lots of name pairs whose degrees of mixed origin vary a lot. The first two corpora were drawn from the Names of The World’s Peoples dictionary (http://zh.wikipedia.org/wiki/世界人名翻译大辞典) published by XinHua Publishing House. The first corpus consists of the names pairs only originating from English language (EO), and the second consists of names originating from English, Chinese, and Japanese evenly (ECJ-O). The third corpus is created by extracting name pairs from LDC (Linguistic Data Consortium) Named Entity List (https://catalog.ldc.upenn.edu/LDC2005T34), which contains names from all over the world (Multi-O). We split the datasets into training, development, and test sets for each corpus randomly with a ratio of 10 : 1 : 1 empirically. The details of three subcorpora are summarized in Table 1.

5.1.2. Baseline System. We compare our alignment model cDPMM with GIZA++ [3] and the Bayesian bilingual alignment model (BBAM) [7]. GIZA++ is one of the most popular alignment models which has been used in machine translation and transliteration frequently with a comparable performance and stability and can be a convincing baseline. BBAM is an alignment model specially for transliteration, and it is the alignment method on each cluster in our cDPMM. By comparing with BBAM the cluster effect caused by our cDPMM will be illustrated.

We employ two decoding models: a phrase-based machine translation decoder (specifically Moses [12]) and the DirecTL decoder [4]. They are based on different decoding strategies and optimization objective which make our comparison more comprehensive.

To evaluate the experimental results, we utilize 3 metrics from the 2012 Named Entities Workshop (NEWS2012) [1]: word accuracy in top-1 (ACC), fuzziness in top-1 (mean F-score), and mean reciprocal rank (MRR).

5.1.3. Hyperparameters Setting. In our model, there are several important hyperparameters to be set by hand:

(i) max $s_i$, the maximum length of the source sequences of a TU;

(ii) max $t_j$, the maximum length of the target sequences of a TU;

(iii) $n_c$, the initial number of clusters for the training data.

We set max $s_i = 6$, max $t_j = 1$, and $n_c = 5$ empirically based on a small pilot experiment. The Moses decoder is used with default settings except for the distortion-limit which is set to 0 to ensure monotonic decoding. For the DirecTL decoder the following settings are used: $cs = 4$, $ng = 9$, and $nBest = 5$. $cs$ denotes the size of context window for features, $ng$ indicates the size of n-gram features, and $nBest$ is the size of transliteration candidate list for updating the model in each iteration. In the BBAM and cDPMM,
Table 1: The size of the three multiorigin corpora.

<table>
<thead>
<tr>
<th>Corpora</th>
<th>Training set</th>
<th>Corpus size</th>
<th>Dev. set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>EO</td>
<td>32,680</td>
<td>3,267</td>
<td>3,267</td>
<td></td>
</tr>
<tr>
<td>ECJ-O</td>
<td>32,500</td>
<td>3,250</td>
<td>3,250</td>
<td></td>
</tr>
<tr>
<td>Multi-O</td>
<td>33,290</td>
<td>3,328</td>
<td>3,328</td>
<td></td>
</tr>
</tbody>
</table>

the concentration parameter $\alpha$ is obtained by sampling its value. Following [17] we use a vague gamma prior $\Gamma(10^{-4}, 10^4)$ and sample new values from a log-normal distribution whose mean is the value of the parameter and variance is 0.3. We use the Metropolis-Hastings algorithm to determine whether this new sample would be accepted. The parameters $\lambda_1$ and $\lambda_2$ in (2) are set to $\lambda_1 = 4$ and $\lambda_2 = 1$.

5.2. Experimental Results

5.2.1. Alignment Results. Table 2 shows some details of the results by different alignment strategies. The #(Targets) represents the average length of English character sequences that are aligned to each Chinese sequence. From the results we can see that in terms of the length of alignment targets: GIZA++ > cDPMM > BBAM. GIZA++ has considerably more targets than the other approaches likely because it overfits the training data. The cDPMM can alleviate the overfitting using its BBAM component and model the diversity caused by multiorigin sequences effectively at the same time.

Table 3 shows some example of TUs segmentation from the alignments produced by BBAM and cDPMM on corpora Multi-O. The information in brackets in Table 3 represents the ID of the cluster and origin of the name pair; the symbol “.” indicates a “NULL” alignment. We can see that the Chinese characters “丁” (ding), “— (yi),” and “ cio (dong)” have different alignments in different origins and that the cDPMM can get the correct alignments.

Figure 2 is the distribution of the size of clustered dataset in the three corpora. The $x$-axis is the range of dataset size; the $y$-axis is the number of clusters. Each column represents the number of clusters which have name pairs and in range of $(a, b]$. Figure 2 shows the cluster size distribution of each corpus. We can see most of the clusters have less than one thousand name pairs, and this may be caused by the fact that some names have strong similarity compared with some other name pairs in the same origin. And we can see that many clusters have less than 10 name pairs because of our sampling strategy. In the following sections, we conduct a different strategy to make use of the cluster and alignment results for training cluster-specific transliteration models.

5.2.2. Evaluation Results of Different Alignment Methods. We run GIZA++ with its default setting as a standard SMT task. For BBAM and cDPMM models, we use the sampled alignment running for 100 iterations which is determined by experience, where every iteration will run about 20 minutes, and combine the alignment tables of each cluster. The experiments are therefore investigating whether the alignment has been improved by the clustering process or not. The top-10 transliteration candidates were used for testing. The detailed experimental results are shown in Tables 4 and 5.

Our cDPMM model achieves the highest performance on all three datasets for all evaluation metrics by a considerable margin. Surprisingly, for dataset EO although there is no multiorigin factor, we still observe a notable improvement. This shows that although names may have monolingual origin, there are possible hidden factors (gender or convention) which can allow our model to fit. Other models based on supervised classification or clustering with fixed classes may fail to capture these characteristics.

To guarantee the reliability of the comparative results, we performed significance testing based on paired bootstrap resampling [26]. We found all differences significant ($p < 0.05$).

5.2.3. Evaluation Results of Cluster Combination Method. In order to make the most use of the cluster results of the cDPMM and exploit the origin information in decoding step, we train a single transliteration model for each cluster. As Figure 2 shows, most of the clusters have a small amount of name pairs and are far from enough to train a good transliteration model. Then, we use the cluster perplexities introduced in Section 4 to combine the small cluster to a larger one which has the highest perplexity, and we make sure each origin has at least one-tenth of the whole training set.

Based on our combined method, we finally got 3 clusters for EO corpus, 6 clusters for ECJ-O corpus, and 4 clusters for Multi-O corpus. As we cluster the name pairs based on the spelling similarity and many factors can cause the spelling differences such as gender and convention, we just assign numbers to the clusters in each corpus based on
Table 3: Segmentation examples from the BBAM and cDPMM.

<table>
<thead>
<tr>
<th>cDPMM alignment</th>
<th>BBAM alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>mun</td>
<td>蒙</td>
</tr>
<tr>
<td>ding</td>
<td>丁</td>
</tr>
<tr>
<td>tei</td>
<td>丁</td>
</tr>
<tr>
<td>fan</td>
<td>范</td>
</tr>
<tr>
<td>hong</td>
<td>洪</td>
</tr>
<tr>
<td>sei</td>
<td>静</td>
</tr>
<tr>
<td>don</td>
<td>东</td>
</tr>
<tr>
<td>he</td>
<td>何</td>
</tr>
<tr>
<td>b</td>
<td>布</td>
</tr>
</tbody>
</table>

Table 4: Comparison of different alignment methods using the Moses.

<table>
<thead>
<tr>
<th>Corpora</th>
<th>Models</th>
<th>ACC</th>
<th>Evaluation M-F-score</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>EO</td>
<td>GIZA++</td>
<td>0.7241</td>
<td>0.8881</td>
<td>0.8061</td>
</tr>
<tr>
<td></td>
<td>BBMM</td>
<td>0.7286</td>
<td>0.8920</td>
<td>0.8043</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.7398</td>
<td>0.8983</td>
<td>0.8126</td>
</tr>
<tr>
<td>ECJ-O</td>
<td>GIZA++</td>
<td>0.5471</td>
<td>0.7278</td>
<td>0.6268</td>
</tr>
<tr>
<td></td>
<td>BBAM</td>
<td>0.5522</td>
<td>0.7370</td>
<td>0.6344</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.5643</td>
<td>0.7420</td>
<td>0.6446</td>
</tr>
<tr>
<td>Multi-O</td>
<td>GIZA++</td>
<td>0.4993</td>
<td>0.4993</td>
<td>0.5986</td>
</tr>
<tr>
<td></td>
<td>BBAM</td>
<td>0.5163</td>
<td>0.7769</td>
<td>0.6123</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.5273</td>
<td>0.7796</td>
<td>0.6188</td>
</tr>
</tbody>
</table>

Table 5: Comparison of different alignment methods using the DirectTL.

<table>
<thead>
<tr>
<th>Corpora</th>
<th>Models</th>
<th>ACC</th>
<th>Evaluation M-F-score</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>EO</td>
<td>GIZA++</td>
<td>0.6950</td>
<td>0.8812</td>
<td>0.7632</td>
</tr>
<tr>
<td></td>
<td>BBMM</td>
<td>0.7152</td>
<td>0.8999</td>
<td>0.7839</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.7231</td>
<td>0.8933</td>
<td>0.7941</td>
</tr>
<tr>
<td>ECJ-O</td>
<td>GIZA++</td>
<td>0.3325</td>
<td>0.6208</td>
<td>0.4064</td>
</tr>
<tr>
<td></td>
<td>BBAM</td>
<td>0.3427</td>
<td>0.6259</td>
<td>0.4192</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.3521</td>
<td>0.6302</td>
<td>0.4316</td>
</tr>
<tr>
<td>Multi-O</td>
<td>GIZA++</td>
<td>0.3815</td>
<td>0.7053</td>
<td>0.4592</td>
</tr>
<tr>
<td></td>
<td>BBAM</td>
<td>0.3934</td>
<td>0.7146</td>
<td>0.4799</td>
</tr>
<tr>
<td></td>
<td>cDPMM</td>
<td>0.3970</td>
<td>0.7179</td>
<td>0.4833</td>
</tr>
</tbody>
</table>

6. Conclusions

In this paper we exploit a coupled Dirichlet process mixture model (cDPMM) to address the overfitting and names multilingual identification issues when transliterating a Western foreign name into Chinese. Furthermore, according to the cluster information generated, we build clustering-specific transliteration models in the decoding step by combining accuracy of each origin-special transliteration model. Table 8 is the comparisons of cDPMM and CCM methods using Moses as the decoder; here “CCM” represents the cluster combination method. From Tables 7 and 8, we can see the CCM get a significant improvement in all metrics of corpora EO and Multi-O. Although in ECJ-O the CCM does not outperform the cDPMM, the performance is still comparable.
small clusters into a large cluster based on the perplexities of name language and transliteration model. Our experiments on the three multiorigin datasets show that the cDPMM model can improve the performance of a transliteration generation system comparing with two other state-of-the-art aligners. And the cluster combination model (CCM) can achieve a significant improvement in two of all three corpora. In the future, other target languages like Japanese and Korean will be tried to test whether our method is language independent or not. And improving the sampling efficiency will be another work as well.

**Conflict of Interests**

The authors declare that there is no conflict of interests regarding the publication of this paper.

**Acknowledgments**

The authors would like to thank their colleagues in HIT for insightful discussions and two anonymous reviewers for many invaluable comments and suggestions to improve their paper. This work is supported by the project of National Natural Science Foundation of China (61173073, 61272384) and International Science and Technology Cooperation Program of China (2014DFA11350).

**References**


Submit your manuscripts at http://www.hindawi.com