Regularized Interlingual Projections: Evaluation on Multilingual Transliteration

Jagadeesh Jagarlamudi
University of Maryland
College Park, USA, 20742
jags@umiacs.umd.edu

Hal Daumé III
University of Maryland
College Park, USA, 20742
hal@umiacs.umd.edu

Abstract

In this paper, we address the problem of building a multilingual transliteration system using an interlingual representation. Our approach uses international phonetic alphabet (IPA) to learn the interlingual representation and thus allows us to use any word and its IPA representation as a training example. Thus, our approach requires only monolingual resources: a *phoneme dictionary* that lists words and their IPA representations.[1] By adding a phoneme dictionary of a new language, we can readily build a transliteration system into any of the existing previous languages, without the expense of all-pairs data or computation. We also propose a regularization framework for learning the interlingual representation, which accounts for language specific phonemic variability, and thus it can find better mappings between languages. Experimental results on the name transliteration task in five diverse languages show a maximum improvement of 29% accuracy and an average improvement of 17% accuracy compared to a state-of-the-art baseline system.

1 Introduction

Because of the wide usage of English, many natural language processing (NLP) tasks have bilingual resources from English into other languages. For example, significantly larger parallel texts are available between English and other languages. Similarly, bilingual dictionaries and transliteration data sets are more accessible from a language into English than into a different language. This situation has caused the NLP community to develop approaches which use a resource rich language (Q say English) as pivot to build resources/applications between a new language pair P and R. Previous studies in machine translation (Utiyama and Isahara, 2007; Paul and Sumita, 2011), transliteration (Khapra et al., 2010), and dictionary mining (Saralegi et al., 2011) show that these bridge language approaches perform competitively with approaches that use resources between P and R. In this paper, we propose a regularization framework for bridge language approaches and show its effectiveness for name transliteration task. The key idea of our approach is that it accounts for language specific variation in the bridge language resources (i.e. between $P \leftrightarrow Q$ and $Q \leftrightarrow R$) and aims to minimize this variation as much as possible. Though our technique is general, for clarity we describe it in the context of named entity (NE) transliteration.

Named entity (NE) transliteration involves transliterating a name in one language into another language and is shown to be crucial for machine translation (MT) (Knight and Graehl, 1998; Al-Onaizan and Knight, 2002; Hermjakob et al., 2008; Li et al., 2009) and cross-lingual information retrieval (CLIR) (AbdulJaleel and Larkey, 2003; Mandl and Womser-Hacker, 2005; Udupa et al., 2009). There exists a large body of literature in transliteration, especially in the bilingual setting, well summarized by Ravi and Knight (2009). We

[1] It is arguable that getting words and their IPA representation require knowledge about both words and IPA symbols, but it still is specific to one language and, in this sense, we refer to it as a monolingual resource.

Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 12–23, Jeju Island, Korea, 12–14 July 2012.

English		Bulgarian	
Word	IPA	Word	IPA
bashful	/ˈbæʃfəl/	шибам	/ˈʃibəm/
tuesday	/ˈtuːzdeɪ/	лук	/luk/
craft	/kɹæft/	как	/kak/
book	/bʊk/	музей	/mʊˈzej/
head	/hɛd/	спека	/spɛˈkɤ/

Table 1: Example phoneme dictionaries in English and Bulgarian. The English translations for the Bulgarian words are switch, onion, how, museum, and spekle.

summarize the approaches that are most relevant to us in Sec. 5. In this paper, we operate in the context of transliteration mining (Klementiev and Roth, 2006; Sproat et al., 2006) where we assume that we are given a source language name and a list of target language candidate transliterations and the task is to identify the correct transliteration.

Given a set of l languages, we address the problem of building a transliteration system between every pair of languages. A straight forward supervised learning approach would require training data of name pairs between every pair of languages (Knight and Graehl, 1998) or a set of common names transliterated from every language into a pivot language. Though it is relatively easy to obtain names transliterated into a pivot language (such as English), it is unlikely that such data sets contain the same names. Bridge language approaches overcome the need for common names and build transliteration systems for resource poor languages (Khapra et al., 2010). However, such approaches still require training data consisting of bilingual name transliterations (orthographic name-to-name mappings). In this paper, we relax the need for name transliterations by using international phonetic alphabet (IPA) in a manner akin to a "bridge language."

2 IPA for Transliteration

We assume that we have a list of words and their IPA representations in each of the l languages. The words in different languages need not have any relationship to each other. Table 1 shows few words and their IPA representations in English and Bulgarian languages. We refer to the set of (word, IPA) pairs as *phoneme dictionary* in this paper. Notice that the common symbols in the IPA sequences indicate a vague phonetic correspondence between the character sequences of English and Bulgarian. For example, both the words 'bashful' and 'шибам' have the symbol 'ʃ' in their IPA sequences which indicate a possible mapping between the character sequences 'sh' and 'ш'.

The use of IPA as the bridge language offers multiple advantages. As shown in Table 1, it allows us to include any (word, IPA) pair in the training data and thus it relaxes the need for name pairs as the training data. Since we only need a phoneme dictionary in each language, our approach does not require any bilingual resources to build the transliteration system. Moreover, since our training data can contain any word (not only the NEs), it is easier to obtain such a resource, for e.g. the phoneme dictionaries obtained from Wiktionary contain at least 2000 words in 21 languages and we will see in Sec. 6 that we can build a decent transliteration system with 2000 words.[2] Finally, unlike other transliteration approaches, by simply adding a phoneme dictionary of $(l+1)^{\text{st}}$ language we can readily get a transliteration system into any of the existing l languages and thus avoid the need for all-pairs data or computation.

Using IPA as the bridge language poses some new challenges such as the language specific phonemic inventory. For example, Mandarin doesn't have /v/, so it is frequently substituted with /w/ or /f/. Similarly, !Xóõ (Southern Khoisan, spoken in Botswana) has 122 consonants, mostly consisting of a large inventory of different word-initial click sounds (Haspelmath et al., 2005), many of which do not exist in any other documented languages. Besides this language specific phonemic inventory, names have different IPA representations in different languages. For example, as shown in Table 2, the IPA sequences for 'China' in English and Dutch have common IPA symbols but the English IPA sequence has additional symbols. Moreover, a name can have multiple pronunciations with in a language, e.g. 'France' has two different IPA sequences in English (Table 2).

In order to handle this phonemic diversity, our method explicitly models language-specific variability and attempts to minimize this phonemic variabil-

[2] In our experiments, we consider languages with small (2000) and big (>30K) phoneme dictionaries.

Word	IPA sequence
China	/'tʃaɪ.nə/ (En), /'ʃina/ (Du), /'çi:na:/ (De)
America	/ə'mɛrɪkə/ (En), /a'me.ri.ka/ (Ro)
France	/'fɹɑ:ns/ (En), /'fɹænts/ (En), /fʁɑ̃s/ (Fr)

Table 2: IPA sequences of few words in different languages indicated using language codes in the parenthesis ('En' for English, 'Du' for Dutch, 'De' for German, 'Ro' for Romanian, and 'Fr' for French).

ity as much as possible. At a high level, our approach uses the phoneme dictionaries of each language to learn mapping functions into an interlingual representation (also referred as common subspace). Subsequently, given a pair of languages, a query name in one of the languages and a list of candidate transliterations in the other language, we use the mapping functions of those two language to identify the correct name transliteration. The mapping functions explicitly model the language specific variability and thus account for fine grained differences. Our experimental results on four language pairs from two different language families show a maximum improvement of 29% accuracy and an average improvement of 17% accuracy compared to a state-of-the-art baseline approach. An important advantage of our approach is that, it extends easily to more than two languages and in fact adding phoneme dictionary from a different, but related, language improves the accuracies of a given language pair. Our main contributions are: 1) building a transliteration system using (word, IPA) pairs and hence using only monolingual resources and 2) proposing a regularization framework which is more general and applies to other bridge language applications such as lexicon mining (Mann and Yarowsky, 2001).

3 Low Dimensional Projections

Our approach is inspired by the Canonical Correlation Analysis (CCA) (Hotelling, 1936) and its application to transliteration mining (Udupa and Khapra, 2010).

First, we convert the phoneme dictionary of each language into feature vectors, i.e. we convert each word into a feature vector of n-gram character sequences and similarly, we also, convert the IPA representations into feature vectors of n-gram IPA symbol sequences. For example, if we use unigram and bigram sequences as features, then the feature vectors of 'head' and its IPA sequence `hɛd' are given by $\{h, e, a, d, \#h, he, ea, ad, d\$\}$ and $\{h$,ɛ, $d, \#h, h$ɛ,ɛ$d, d\$\}$. For brevity, we refer to the spaces of n-gram character and IPA symbol sequences as character and phonemic spaces respectively. The character space is specific to each language while the phonemic space is shared across all the languages. Since we use IPA as bridge, even though two languages share orthography (e.g. English and French) it is irrelevant for our approach.

Then, for each language, we find mappings $\big(A_i$ and $U_i\big)$ from the character and phonemic spaces into a common k-dimensional subspace such that the correct transliterations lie closer to each other in this subspace. Before moving into the details of our approach, we will describe the notation and then give an overview of the process by which our approach finds the transliteration.

3.1 Notation

Let $\mathbf{x}_i^{(m)} \in \mathbb{R}^{d_i}$ and $\mathbf{p}_i^{(m)} \in \mathbb{R}^c$ be the feature vectors of the m^{th} word and its IPA sequence in the $i^{th}\big(1 \cdots l\big)$ language, where d_i is the size (i.e. no. of features) of the character space of the language and c is the size of the common phonemic space. Let X_i $(d_i \times n_i)$ and P_i $(c \times n_i)$ denote the i^{th} language data matrices with $\mathbf{x}_i^{(m)}$ and $\mathbf{p}_i^{(m)}$ $m = 1 \cdots n_i$ as the columns respectively. We consistently use subscript to indicate the language and superscript to indicate the index of an example point.

3.2 Method Overview

During the training stage, for each language, we find mappings (or projection directions) $A_i \in \mathbb{R}^{(d_i \times k)}$ and $U_i \in \mathbb{R}^{(c \times k)}$ from the character and phonemic spaces into a k-dimensional subspace (or an interlingual representation) such that a name gets mapped to the *same* k-dimensional vector irrespective of the language. That is, given a name $\mathbf{x}_i$ it gets mapped to the vector $A_i^T \mathbf{x}_i$ and similarly its IPA sequence $\mathbf{p}_i$ gets mapped to $U_i^T \mathbf{p}_i$. During the testing stage, given a name $\mathbf{x}_i$ in the source (i^{th}) language, we find its transliteration in the target (j^{th}) language $\mathbf{x}_j$ by solving the following decoding problem:

$$\arg\min_{\mathbf{x}_j} L\big(\mathbf{x}_i, \mathbf{x}_j\big) \tag{1}$$

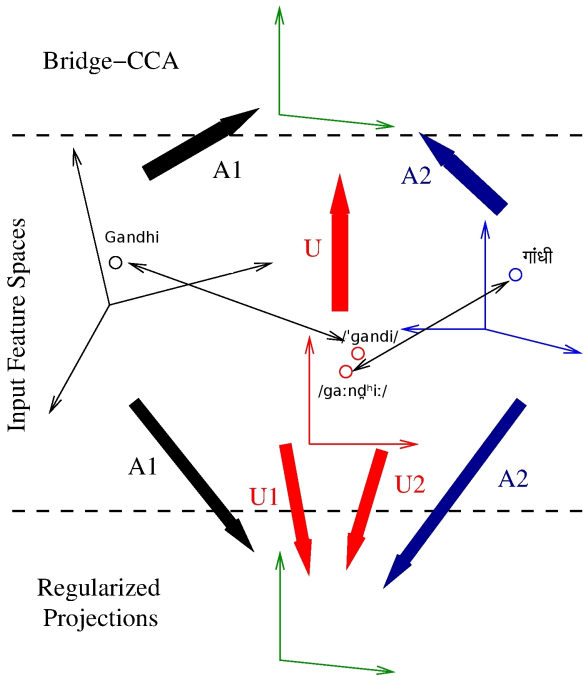

Figure 1: A single name (Gandhi) is shown in all the input feature spaces. The alignment between the character and phonemic space is indicated with double dimensional arrows. Bridge-CCA uses a single mapping function U from the phonemic space into the common subspace (the 2-dimensional green space at the top), where as our approach uses two mapping functions U_1 and U_2, one for each language, to map the IPA sequences into the common subspace.

where $L(\mathbf{x}_i, \mathbf{x}_j)$ is given by

$$\min_{\mathbf{p} \in \mathbb{R}^c} \|A_i^T \mathbf{x}_i - U_i^T \mathbf{p}\|^2 + \|A_j^T \mathbf{x}_j - U_j^T \mathbf{p}\|^2 \quad (2)$$

This formulation uses the source language mappings (A_i and U_i) to find the IPA sequence $\mathbf{p}$ that is closest to the source name and then uses it, along with the target language mappings (A_j and U_j), to identify the correct transliteration from a list of candidate transliterations.

At a high level, existing bridge language approaches such as Bridge-CCA (Khapra et al., 2010) assume that $U_i \equiv U_j$ thus ignoring the language specific variation. To understand its implication consider the example shown in Fig. 1. The middle portion of the Fig. shows the name Gandhi (represented as point) in the character spaces of English and Hindi, three-dimensional spaces, and its IPA sequences in the phonemic space (the two-dimensional space in the middle). Notice that, because of the phonemic variation, the same name is represented by two distinct points in the common phonemic space.[3] Now, since Bridge-CCA uses a single mapping function for both the IPA sequences, it fails to map these two distinct points into a common point in the interlingual subspace.

Our new formulation, as explained above, relaxes this hard constraint and learns different mapping functions (U_i and U_j) and hence our approach can potentially map both the distinct IPA sequences into a single point. As a result our approach successfully handles the language specific phonemic variation. At the same time we constrain the projection directions such that they behave similarly for the phonemic sounds that are observed in majority of the languages. In the example shown in Fig. 1, our model (called Regularized Projections) finds two different mapping functions U_1 and U_2, one for each language, from the phonemic space into the common two-dimensional space at the bottom.

3.3 Regularized Projections

In this section we first formulate the problem of finding the mapping functions (A_i and U_i) of each language as an optimization problem. In the following section (Sec. 4), we develop a method for solving the optimization problem and also derive closed form solution for the prediction problem given in Eq. 1. For simplicity, we describe our approach in terms of single projection vectors, $\mathbf{a}_i \in \mathbb{R}^{d_i}$ and $\mathbf{u}_i \in \mathbb{R}^c$, rather than full matrices, but the generalization is trivial.

Inspired by the Canonical Correlation Analysis (CCA) (Hotelling, 1936), we find projection directions in the character and phonemic spaces of each language such that, after projection, a word is closer to its aligned IPA sequence. To understand this, assume that we have a name (say "Barack Obama") in all the languages[4] and its feature vectors are given by $\mathbf{x}_i$ and $\mathbf{p}_i$ $i = 1 \cdots l$ in the character and phone-

[3] In reality, as explained in the previous section, the phonemic variation that is commonly observed is that different features are triggered for different languages. But for visualization purpose, we showed the IPA sequences as if they differ in the feature values.

[4] Our model does not require same names in different languages; this is used only for easier understanding.

mic spaces respectively. Then, we might try to find projection directions $\mathbf{a}_i$ in each language and $\mathbf{u}$ in the common phonemic space such that:

$$\arg\min_{\mathbf{a}_i,\mathbf{u}} \sum_{i=1}^{l} \Big(\langle \mathbf{x}_i, \mathbf{a}_i \rangle - \langle \mathbf{p}_i, \mathbf{u} \rangle \Big)^2 \qquad (3)$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product between two vectors. This model assumes that the projection direction $\mathbf{u}$ is same for the phonemic space of all the languages. This is a hard constraint and does not handle the language specific variability as discussed in the previous section. We model the language specificity by relaxing this hard constraint.

In our model, intuitively, the parameters corresponding to the phonemic sounds that occur in majority of the languages are shared across the languages while the parameters of the language specific sounds are modeled per each language. This is achieved by modeling the projection directions of the i^{th} language phonemic space $\mathbf{u}_i \leftarrow \mathbf{u} + \mathbf{r}_i$. The vector $\mathbf{u} \in \mathbb{R}^c$ is common to the phonemic spaces of all the languages and thus handles sounds that are observed in multiple languages while $\mathbf{r}_i \in \mathbb{R}^c$, the residual vector, is specific to each language and accounts for the language specific phonemic variations. Then the new formulation is given by:

$$\arg\min_{\mathbf{a}_i,\mathbf{u},\mathbf{r}_i} \sum_{i=1}^{l} \|\langle \mathbf{x}_i, \mathbf{a}_i \rangle - \langle \mathbf{p}_i, \mathbf{u} + \mathbf{r}_i \rangle\|^2 + \lambda \langle \mathbf{p}_i, \mathbf{r}_i \rangle^2$$

where λ is the residual parameter. The first term of this summation ensures that a word and its IPA sequence gets mapped to closer points in the subspace while the second term forces the residual vectors to be as small as possible. By enforcing the residual vectors to be small, this formulation encourages the sounds that occur in majority of the languages to be accounted by $\mathbf{u}$ and the sounds that are specific to the given language by $\mathbf{r}_i$. The final optimization problem is obtained by summing these terms over all the examples and all the languages and is given by:

$$\min_{\mathbf{a}_i,\mathbf{u},\mathbf{r}_i} \sum_{i=1}^{l} \Big(\frac{||X_i^T \mathbf{a}_i - P_i^T(\mathbf{u} + \mathbf{r}_i)||^2}{n_i} + \lambda \, \|P_i^T \mathbf{r}_i\|^2 \Big) \quad (4)$$

$$\text{s.t.} \sum_{i=1}^{l} \frac{1}{n_i} \|X_i^T \mathbf{a}_i\|^2 = 1 \text{ and } \sum_{i=1}^{l} \frac{1}{n_i} \|P_i^T \mathbf{u}\|^2 = 1$$

The constraints of the above optimization problem avoid the trivial solution of setting all the vectors to zero and are referred to as length constraints.

4 Model Optimization

In this section, we derive the solutions for the optimization problems presented in the previous section.

4.1 Training the Model

We follow the standard procedure of forming the Lagrangian and setting its derivative to zero. The Lagrangian $\mathcal{L}$ of the optimization problem in Eq. 4 is given by:

$$\mathcal{L} = \sum_i \frac{1}{n_i} ||X_i^T \mathbf{a}_i - P_i^T(\mathbf{u} + \mathbf{r}_i)||^2 + \lambda \sum_i \|P_i^T \mathbf{r}_i\|^2$$

$$+\alpha \Big(\sum_i \frac{1}{n_i} \|X_i^T \mathbf{a}_i\|^2 - 1 \Big) + \beta \Big(\sum_i \frac{1}{n_i} \|P_i^T \mathbf{u}\|^2 - 1 \Big)$$

where α and β are Lagrangian multipliers corresponding to the length constraints. Differentiating $\mathcal{L}$ with respect to $\mathbf{a}_i$, $\mathbf{r}_i$ and $\mathbf{u}$ and setting the derivatives to zero yields the following equations, respectively:

$$\begin{aligned} (1+\alpha) X_i X_i^T \mathbf{a}_i - X_i P_i^T \mathbf{r}_i &= X_i P_i^T \mathbf{u} \\ -P_i X_i^T \mathbf{a}_i + (1 + \lambda n_i) P_i P_i^T \mathbf{r}_i &= -P_i P_i^T \mathbf{u} \end{aligned}$$

$$\sum_i \frac{1}{n_i} \Big(P_i X_i^T \mathbf{a}_i - P_i P_i^T \mathbf{r}_i \Big) = (1+\beta) \sum_i \frac{1}{n_i} P_i P_i^T \mathbf{u}$$

We can rewrite these equations in matrix form, as shown in Eqs. 5 and 6, since the solution becomes clear in this form. For brevity, let $E_i = (1 + \alpha) X_i X_i^T$, $F_i = -X_i P_i^T$ and $G_i = (1 + \lambda n_i) P_i P_i^T$. Then, $\mathbf{u}$ can be solved for using the generalized eigenvalue problem shown in Eq. 7. This step involves computing an inverse of a $(d_i + c)$ matrix and an eigenvalue problem of size c which can be expensive since solving each of these problems involve cubic time. This can be reduced further into a problem of smaller size by using inverse of a partitioned matrix as shown in Eq. 8. This identity reduces the matrix inverse computation from a problem of size $d_i + c$ into two smaller problems of size d_i and c each. This reduces the time complexity considerably since the inverse computation is cubic in the size of the matrix.

$$\begin{bmatrix}(1+\alpha)X_iX_i^T & -X_iP_i^T \\ -P_iX_i^T & (1+\lambda n_i)P_iP_i^T\end{bmatrix}\begin{bmatrix}\mathbf{a}_i \\ \mathbf{r}_i\end{bmatrix} = \begin{bmatrix}X_iP_i^T \\ -P_iP_i^T\end{bmatrix}\mathbf{u} \quad (5)$$

$$\sum_i \frac{1}{n_i}\begin{bmatrix}P_iX_i^T & -P_iP_i^T\end{bmatrix}\begin{bmatrix}\mathbf{a}_i \\ \mathbf{r}_i\end{bmatrix} = (1+\beta)\sum_i \frac{1}{n_i}P_iP_i^T\mathbf{u} \quad (6)$$

$$\sum_i \frac{1}{n_i}\begin{bmatrix}-F_i^T & \frac{-G_i}{1+\lambda n_i}\end{bmatrix}\begin{bmatrix}E_i & F_i \\ F_i^T & G_i\end{bmatrix}^{-1}\begin{bmatrix}-F_i \\ \frac{-G_i}{1+\lambda n_i}\end{bmatrix}\mathbf{u} = (1+\beta)\sum_i \frac{1}{n_i}P_iP_i^T\mathbf{u} \quad (7)$$

$$\text{If } M_i = \left(E_i - F_iG_i^{-1}F_i^T\right)^{-1}, \text{ then } \begin{bmatrix}E_i & F_i \\ F_i^T & G_i\end{bmatrix}^{-1} = \begin{bmatrix}M_i & -M_iF_iG_i^{-1} \\ -G_i^{-1}F_i^TM_i & G_i^{-1}+G_i^{-1}F_i^TM_iF_iG_i^{-1}\end{bmatrix} \quad (8)$$

Substituting Eq. 8 into Eq. 7 and further simplifying results in the following eigenvalue problem for solving $\mathbf{u}$:

$$\sum_i \frac{G_i + (\lambda n_i)^2F_i^TM_iF_i}{n_i(1+\lambda n_i)^2}\mathbf{u} = (1+\beta)\sum_i \frac{P_iP_i^T}{n_i}\mathbf{u}$$

where $M_i = \left(E_i - F_iG_i^{-1}F_i^T\right)^{-1}$. Notice that the term $E_i = (1+\alpha)X_iX_i^T$ depends on the Lagrangian multiplier α. Because of this, we cannot solve for both the parameters and the Lagrangian multipliers at the same time. One possible approach is to do an alternate optimization over the parameters and Lagrangian multipliers, but in this paper we fix α and solve for $\mathbf{u}$. The value of α denotes the correlation and its maximum value is 1. In practice, we often observe that the top few correlations take the value of 1. Based on this observation we fix the value of α to 1 (Sec. 6).

Subsequently, we use $\mathbf{u}$ to solve for $\mathbf{a}_i$ and $\mathbf{r}_i$ as follows:

$$\mathbf{a}_i = -\frac{\lambda n_iM_iF_i}{1+\lambda n_i}\mathbf{u} \quad (9)$$

$$\mathbf{r}_i = \frac{\lambda n_iG_i^{-1}F_i^TM_iF_i - I}{1+\lambda n_i}\mathbf{u} \quad (10)$$

In order to increase the stability of the system we regularize G_i and E_i by adding τI. We use the top k eigenvectors $\mathbf{u}$ and their corresponding $\mathbf{a}_i$ and $\mathbf{r}_i$ vectors as columns and form the mappings U, A_i and R_i respectively. These mappings are used in predicting the transliteration of a name in one language into any other language, which will be described in the following section.

4.2 Transliteration Mining (Prediction)

During the testing phase, given a source name and a list of candidate transliterations, we solve the decoding problem shown in Eq. 1 to find the appropriate target language transliteration. Formally, given a word $\mathbf{x}_i$ in i^{th} language we find its transliteration into j^{th} language $\mathbf{x}_j$, by solving the optimization problem shown in Eq. 1, where $U_i = U + R_i$ and $U_j = U + R_j$. Similar to the previous case, the closed form solution can be found by computing the first derivative with respect to the unknown phoneme sequence and the target language transliteration and setting it to zero. First, the IPA sequence $\mathbf{p}^*$ that minimizes $L\left(\mathbf{x}_i, \mathbf{x}_j\right)$ is given by:

$$\mathbf{p}^* = C_{ij}^{-1}\left(U_iA_i^T\mathbf{x}_i + U_jA_j^T\mathbf{x}_j\right) \quad (11)$$

where $C_{ij} = U_iU_i^T + U_jU_j^T$. We substitute this back in Eq. 2 and then solve for $\mathbf{x}_j$, the best transliteration in the j^{th} language, as:

$$A_j\left(I - U_j^TC_{ij}^{-1}U_j\right)A_j^T\mathbf{x}_j = A_jU_j^TC_{ij}^{-1}U_iA_i^T\mathbf{x}_i \quad (12)$$

Since U_i and U_j are not full rank matrices, to increase the numerical stability of the prediction step, we use $C_{ij} = U_iU_i^T + U_jU_j^T + 0.001\,I$ where I is an identity matrix. Notice that this solution doesn't depend on the $\mathbf{p}^*$ and hence we don't need to compute it explicitly.

5 Related Work

There is a large body of the literature in named entity transliteration, so we will describe only the most relevant ones to our approach. In transliteration, generative approaches aim to generate the target language

transliteration of a given source name (Knight and Graehl, 1998; Jung et al., 2000; Haizhou et al., 2004; Al-Onaizan and Knight, 2002) while discriminative approaches assume a list of target language names, obtained from other sources, and try to identify the correct transliteration (Klementiev and Roth, 2006; Sproat et al., 2006). The effectiveness of the discriminative approaches depend on the list of target language candidates. Sproat et al. (2006) report an oracle accuracy of 85%, but it depends on the source of the candidate transliterations. Nevertheless, all these approaches require either bilingual name pairs or phoneme sequences to learn to transliterate between two languages. Thus, if we want to build a transliteration system between every pair of languages in a given set of languages then these approaches need resources between every pair of languages which can be prohibitive.

Bridge language approaches propose an alternative and use a resource rich language such as English as common language (Khapra et al., 2010) but they still need bilingual resources. Moreover BridgeCCA (Khapra et al., 2010) uses a single mapping function for the phonemic space of all the languages and thus it can not handle language specific variability. In the original setting, authors use English as the pivot and since the feature space of English is fixed, irrespective of the target language, this may not be a serious concern but it becomes crucial when we use IPA as the bridge language.

Approaches that map words in different languages into the common phonemic space have also been well studied. But most of these approaches use language specific resources such as CMU pronunciation dictionary (Gao et al., 2004) or a carefully constructed cost matrices for addition, substitution, and deletion of phonemes between a pair of languages (Tao et al., 2006; Yoon et al., 2007). Variants of soundex algorithm (Odel and Russel, 1918) such as Kodex (Kang and Choi, 2000) use hand constructed consonant to soundex code tables for name transliteration. Similar to our approach these variants only require soundex mappings of a new language to build transliteration system, but our model does not require explicit mapping between n-gram characters and the IPA symbols instead it learns them automatically using phoneme dictionaries. Alternatively unsupervised approaches have also been explored (Ravi and Knight, 2009), but their accuracies are fairly low compared to the supervised and weekly supervised approaches.

6 Experiments

Our experiments are designed to evaluate the following three aspects of our model, and of our approach to transliteration in general:

IPA as bridge: Unlike other phonemic based approaches (Sec. 5), we do not *explicitly* model the phoneme modifications between pairs of languages. Moreover, the phoneme dictionary in each language is crawled from Wiktionary (Sec. 6.1), which is likely to be noisy. So, the first aspect we want to evaluate is the effectiveness of using IPA as the bridge language. Here, we also compare our method with other bridge language approaches and establish the importance of modeling language specific variance.

Multilinguality: In our method, simply adding a phoneme dictionary of a new language allows us to extend our transliteration system into any of the existing languages. We evaluate the effect of data from a different, but related, languages on a transliteration system between a given pair.

Complementarity: Using IPA as bridge language allows us to build transliteration system into resource poor languages. But we also want to evaluate whether such an approach can help improving a transliteration system trained directly on bilingual name-pairs.

6.1 Data Sets

We use data sets from five languages in order to evaluate the effectiveness of our approach. The phoneme dictionaries (list of words and their IPA representations as shown in Table 1) are obtained from Wiktionary. The Wiktionary dump downloaded in October 2011 has at least 2000 (word, IPA) pairs in 21 languages which also includes some resource poor languages (e.g. Armenian, Taiwanese, Turkish, etc.).

In principle, our method allows us to build transliteration system into any of these language pairs without any additional information. But, in this paper, we use English (En), Bulgarian (Bg), Russian (Ru), French (Fr), and Romanian (Ro) for eval-

	En.	Bg.	Ru.	Fr.	Ro.
Train	31K	36K	1141	36K	5211
Dev.	–	1264	2000	2717	430
Test	–	1264	2000	2717	431
# Features	5000	3998	2900	5000	3465

Table 3: Statistics of different data sets. Training data is monolingual phoneme dictionaries while development/test sets are bilingual name pairs between English and the respective language.

uation purposes, as they suffice to showcase all the three aspects mentioned in the previous section. Table 3 shows the sizes of phoneme dictionaries used for training the models. The phoneme dictionaries of English, Bulgarian, and Russian contain more than 30K (word,IPA) pairs while the remaining two languages have smaller phoneme dictionaries. The development and test sets between English and the remaining language pairs are obtained from geonames data base.[5] These are geographic location names from different countries written in multiple languages.

6.2 Experimental Setup

We convert the phoneme dictionaries of each language into feature vectors. We use unigram and bigram features in the phonemic space and unigram, bigram and trigram features in the character space. An example for feature generation is shown in Sec. 3. After converting the data into feature vectors, we retain the most frequent 5000 features. We only keep the frequent 5000 features since we observed, elsewhere, that including infrequent features leads CCA based methods to learn projection directions with perfect correlations which are not effective for downstream applications. The last row of Table 3 shows the number of features in the character space of each of the languages. The phonemic space is common to all the languages and has 3777 features. Though the phonemic features are common to all the languages, as discussed in Sec. 2, only a subset of features will be observed in a given language. For example, in our data sets, of the total 3777 common phonetic features only 3312, 882, and 1009 features are observed in English, Bulgarian, and Russian languages respectively. This indicates the diversity in the phonemic inventory of different languages.

[5] http://www.geonames.org/

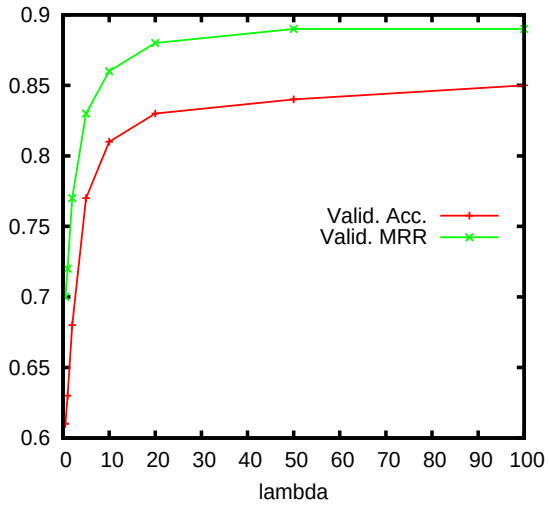

Figure 2: Performance of transliteration system with residual parameter λ on English-Bulgarian development data set.

We compare our approach against Bridge-CCA, a state-of-the-art bridge language transliteration system which is known to perform competitively with other discriminative approaches (Khapra et al., 2010). We use the phoneme dictionaries in each language to train our approach, as well as the baseline system. The projection directions learnt during the training are used to find the transliteration for a test name as described in Sec. 4.2. We report the performance in terms of the accuracy (exact match) of the top ranked transliteration and the mean reciprocal rank (MRR) of the correct transliteration. We find transliterations in both the directions (i.e. target language transliterations given a source name and vice versa) and report average accuracies. The regularization parameter (τ) and the size of the interlingual representation (k) in both our approach and Bridge-CCA are tuned on the development set.

6.3 Description of Results

In this section we report experimental results on the three aspects mentioned above.

6.3.1 IPA as Bridge

Fig. 2 shows the performance of our system with the residual parameter λ (in Eq. 4) on the develop-

		En-Bg		En-Ru		En-Fr		En-Ro	
		Acc.	MRR	Acc.	MRR	Acc.	MRR	Acc.	MRR
1	Bridge-CCA	68.83	77.22	44.50	53.22	41.55	52.89	71.69	79.59
2	Ours (cosine)	67.68	76.52	45.07	53.63	42.45	53.06	74.13	81.28
3	Ours (Eq. 12)	83.70	88.32	63.47	73.01	70.68	78.13	77.38	**84.22**
4	Ours (cosine + Multi.)	68.91	77.44	49.15	57.20	42.55	53.02	77.49	84.04
5	Ours (Eq. 12 + Multi.)	**84.45**	**88.43**	**66.70**	**75.85**	**71.09**	**78.43**	**77.49**	84.04

Table 4: Results of our approach and the baseline system on the test set. The second block shows the results when our approach is trained only on phoneme dictionaries of the language pair, the third block shows results when we include other language data as well.

ment data set. When λ is small, the model does not attempt to constrain the projection directions U_i's and hence they tend to map names to completely unrelated vectors. As we increase the residual parameter, it forces the residual vectors (R_i) to be smaller and thus the subspaces identified for each language are closely tied together. Thus, it models the commonalities across languages and also the language specific variability. Based on the performance curves on the development data, we fix $\lambda = 50$ in the rest of the experiments.

Table 4 shows the results of Bridge-CCA and our approach on the four language pairs. We report the results of our approach with the decoding proposed in Sec. 4.2 and a simple cosine similarity measure in the common-subspace, i.e. $\cos\left(A_i^T\mathbf{x}_i, A_j^T\mathbf{x}_j\right)$. Comparison of the accuracies in rows 1, 2 and 3, shows that simply using cosine similarity performs almost same as the Bridge-CCA approach. However, using the decoding suggested in Eq. 12 gives significant improvements. To understand why the cosine angle between $A_i^T\mathbf{x}_i$ and $A_j^T\mathbf{x}_j$ is not the appropriate measure, assume that the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are feature vectors of same name in two languages and let $\mathbf{p}$ be its true IPA representation. Then, since our model learns projection directions such that $A_i^T\mathbf{x}_i \approx U_i^T\mathbf{p}$,

$$\cos(A_i^T\mathbf{x}_i, A_j^T\mathbf{x}_j) = \cos\left((U+R_i)^T\mathbf{p}, (U+R_j)^T\mathbf{p}\right)$$

The additional residual matrices R_i and R_j make the cosine measure inappropriate. At the same time, our model forces the residual matrices to be small and this is probably the reason why it performs competitively with the Bridge-CCA. On the other hand, our decoding method, as shown in Eq. 1, integrates over the best possible phoneme sequence and thus yields significant improvements. In the rest of the paper, we report results with the decoding in Eq. 12 unless specified explicitly. Our approach achieves a maximum improvement of 29.13% accuracy over Bridge-CCA in English-French and on an average it achieves 17.17% and 15.19% improvement in accuracy and MRR respectively. Notice that even though our Russian phoneme dictionary has only 1141 (word, IPA) pairs, our approach is able to achieve an accuracy of 63.47% and an MRR of 73% indicating that the correct name transliteration is, on an average, at rank 1 or 2.

6.3.2 Multilinguality

The fourth and fifth rows of Table 4 also show the multilingual results. In particular, we train our system on data from the three languages En, Bg, and Ru and test it on En-Bg and En-Ru test sets. Similarly, we train a different system on data from En, Fr and Ro and evaluate it on En-Fr and En-Ro test sets. We split the languages based on the language family, Russian and Bulgarian are Slavonic languages while French and Romanian are Romance languages, and expect that languages in same family have similar pronunciations. Comparing the performance of our system with and without the multilingual data set, it is clear that having data from other languages helps improve the accuracy.

6.3.3 Complementarity

In the final experiment, we want to compare the performance of our approach, which uses only monolingual resources, with a transliteration system trained using bilingual name pairs. We train a CCA based transliteration system (Udupa and Khapra,

	En-Bg		En-Fr	
	Acc.	MRR	Acc.	MRR
CCA	95.57	96.76	95.82	96.67
Ours+CCA	95.69	96.90	96.14	96.90
Δ Err	2.7%	4.2%	7.5%	6.8%
Ours+CCA(t)	95.80	96.95	96.34	97.04
Δ Err	5.4%	5.8%	12.3%	11.3%

Table 5: Comparison with a system trained on bilingual name pairs. The (t) in the third row indicates parameters are tuned for test set. We also show the percentage error reduction achieved by a linear combination of our approach and CCA.

2010) on a training data of 3792 and 8151 location name pairs. Notice that the training and test data for this system are from the same domain and thus it has an additional advantage over our approach, which is trained on whatever happens to be on Wiktionary.

The second row of Table 5 shows the results of CCA on English-Bulgarian and English-French language pairs. CCA achieves high accuracies even though the training data is relatively small, most likely because of the domain match between training and test data sets. As another baseline, we tried using Google machine translation API to transliterate the English names of the En-Bg test set. We hoped that since these are names, the translation engine would simply transliterate them and return the result. Of the output, we observed that about 500 names are passed through the MT system unchanged and so we ignore them. On the remaining names, it achieved an accuracy of 76.15% and the average string edit distance of the returned transliteration to the true transliteration is about 3.74. These accuracies are not directly comparable to the results shown in Table 5 because, presumably, it is a transliteration generation system unlike CCA which is a transliteration mining approach. For lack fair comparison, we don't report the accuracies of the Google transliteration output in Table 5.

Table 5 also shows the results of our system when combined with the CCA approach. For a given English word, we score the candidate transliterations using our approach and then linearly combine their scores with the scores assigned by CCA. We perform a line search between $[0, 1]$ for the appropriate weight combination. The third and fourth rows of Table 5 show the results of the linear combination when the weight is tuned for the development and test sets respectively. The improvements, though not significant, are encouraging and suggest that a sophisticated way of combining these different systems may yield significant improvements. This experiment shows that a transliteration system trained on word and IPA representations can actually augment a system trained on bilingual name pairs leading to an improved performance.

7 Conclusion

In this paper we proposed a regularization technique for the bridge language approaches and showed its effectiveness on the name transliteration task. Our approach learns interlingual representation using only monolingual resources and hence can be used to build transliteration system between resource poor languages. We show that, by accounting the language specific phonemic variation, we can get a significant improvements. Our experimental results suggest that a transliteration system built using IPA data can also help improve the accuracy of a transliteration system trained on bilingual name pairs.

Thought we used IPA as a bridge language there are other viable options. For example, as shown in Khapra et al. (2010) we can use English as the bridge language. Since name transliteration problem is being studied for a considerable time, many resources already exist between English and other languages. So, one can argue the appropriateness of IPA as bridge language compared to, say, English. While this is an important question, in this paper, we are primarily interested in showing the importance of handling language specific phenomenon in the bridge language approaches. In future, we would like to study the appropriateness of IPA vs. English as the bridge language and also the generalizability of our technique to other scenarios.

Acknowledgements

This work is partially funded by NSF grant IIS-1153487 and the BOLT program of the Defense Advanced Research Projects Agency, Contract No. HR0011-12-C-0015.

References

Nasreen AbdulJaleel and Leah S. Larkey. 2003. Statistical transliteration for english-arabic cross language information retrieval. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, pages 139–146, New York, NY, USA. ACM.

Yaser Al-Onaizan and Kevin Knight. 2002. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, SEMITIC '02, pages 1–13, Stroudsburg, PA, USA. ACL.

Wei Gao, Kam fai Wong, and Wai Lam. 2004. Phoneme-based transliteration of foreign names for OOV problem. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP)*, pages 374–381.

Li Haizhou, Zhang Min, and Su Jian. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. ACL.

Martin Haspelmath, Matthew Dryer, David Gil, and Bernard Comrie, editors. 2005. *The World Atlas of Language Structures*. Oxford University Press.

Ulf Hermjakob, Kevin Knight, and Hal Daumé III. 2008. Name translation in statistical machine translation - learning when to transliterate. In *Proceedings of ACL-08: HLT*, pages 389–397, Columbus, Ohio, June. ACL.

Harold Hotelling. 1936. Relation between two sets of variables. *Biometrica*, 28:322–377.

Sung Young Jung, SungLim Hong, and Eunok Paek. 2000. An english to korean transliteration model of extended markov window. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, COLING '00, pages 383–389, Stroudsburg, PA, USA. ACL.

Byung-Ju Kang and Key-Sun Choi. 2000. Two approaches for the resolution of word mismatch problem caused by english words and foreign words in korean information retrieval. In *Proceedings of the 5th international workshop on on Information retrieval with Asian languages*, IRAL '00, pages 133–140, New York, NY, USA. ACM.

Mitesh M. Khapra, Raghavendra Udupa, A. Kumaran, and Pushpak Bhattacharyya. 2010. Pr + rq $\approx$ pq: Transliteration mining using bridge language. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010*, Atlanta, Georgia, USA, July. AAAI Press.

Alexandre Klementiev and Dan Roth. 2006. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 817–824, Stroudsburg, PA, USA. ACL.

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

Haizhou Li, A. Kumaran, Vladimir Pervouchine, and Min Zhang. 2009. Report of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 1–18, Stroudsburg, PA, USA. ACL.

Thomas Mandl and Christa Womser-Hacker. 2005. The effect of named entities on effectiveness in cross-language information retrieval evaluation. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 1059–1064, New York, NY, USA. ACM.

Gideon S. Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Proceedings of the 2nd meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA. ACL.

M. K. Odel and R. C. Russel. 1918. U.s. patent numbers, 1,261,167 (1918) and 1,435,663(1922).

Michael Paul and Eiichiro Sumita. 2011. Translation quality indicators for pivot-based statistical mt. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 811–818, Chiang Mai, Thailand, November. AFNLP.

Sujith Ravi and Kevin Knight. 2009. Learning phoneme mappings for transliteration without parallel data. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 37–45, Boulder, Colorado, June. ACL.

Xabier Saralegi, Iker Manterola, and Iñaki San Vicente. 2011. Analyzing methods for improving precision of pivot based bilingual dictionaries. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 846–856, Edinburgh, Scotland, UK., July. ACL.

Richard Sproat, Tao Tao, and ChengXiang Zhai. 2006. Named entity transliteration with comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 73–80, Stroudsburg, PA, USA. ACL.

Tao Tao, Su-Youn Yoon, Andrew Fister, Richard Sproat, and ChengXiang Zhai. 2006. Unsupervised named

entity transliteration using temporal and phonetic correlation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 250–257, Stroudsburg, PA, USA. ACL.

Raghavendra Udupa and Mitesh M. Khapra. 2010. Transliteration equivalence using canonical correlation analysis. In *ECIR'10*, pages 75–86.

Raghavendra Udupa, Saravanan K, Anton Bakalov, and Abhijit Bhole. 2009. "they are out there, if you know where to look": Mining transliterations of oov query terms for cross-language information retrieval. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 437–448, Berlin, Heidelberg. Springer-Verlag.

Masao Utiyama and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 484–491, Rochester, New York, April. ACL.

Su-Youn Yoon, Kyoung-Young Kim, and Richard Sproat. 2007. Multilingual transliteration using feature based phonetic method. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 112–119, Prague, Czech Republic, June. ACL.