Leveraging Entity Linking and Related Language Projection to Improve Name Transliteration

Ying Lin[1], **Xiaoman Pan**[1], **Aliya Deri**[2], **Heng Ji**[1], **Kevin Knight**[2]
[1] Computer Science Department, Rensselaer Polytechnic Institute
`{liny9,panx2,jih}@rpi.edu`
[2] Information Sciences Institute, University of Southern California
`{aderi,knight}@isi.edu`

Abstract

Traditional name transliteration methods largely ignore source context information and inter-dependency among entities for entity disambiguation. We propose a novel approach to leverage state-of-the-art Entity Linking (EL) techniques to automatically correct name transliteration results, using collective inference from source contexts and additional evidence from knowledge base. Experiments on transliterating names from seven languages to English demonstrate that our approach achieves 2.6% to 15.7% absolute gain over the baseline model, and significantly advances state-of-the-art. When contextual information exists, our approach can achieve further gains (24.2%) by collectively transliterating and disambiguating multiple related entities. We also prove that combining Entity Linking and projecting resources from related languages obtained comparable performance as the method using the same amount of training pairs in the original languages without Entity Linking.[1]

1 Introduction

In Machine Translation and Cross-lingual Information Extraction tasks, an important problem is translating out-of-vocabulary words, mostly names. For some names, we can perform transliteration (Knight and Graehl, 1997; Knight and Graehl, 1998), namely converting them to their approximate phonetic equivalents. Previous methods have generally followed the two-step approach proposed by (Al-Onaizan and Knight, 2002): Generating transliteration hypotheses based on phoneme, grapheme or correspondence, and validating or re-ranking hypotheses using language modeling (Oh and Isahara, 2007) or Information Extraction from the target language (Ji et al., 2009).

In this paper, we focus on *back*-transliteration from languages lacking in Natural Language Processing (NLP) resources to English for two reasons: (1) In NLP tasks such as name tagging, we can take advantage of rich English resources by transliterating a name to English. Our analysis of 986 transliteration pairs from the Named Entities Workshop 2015 (NEWS2015) [2] Bengali development set shows that 574 English names can be found in the DBpedia [3], while only 47 Bengali names exist in the same knowledge base (KB). (2) *Back*-transliterating names in other languages to English make them understandable by more users since English is widely spoken as a global lingua franca.

In this paper we analyze the following remaining challenges from previous methods:

Challenge 1: Lack of Entity Grounding. Previous methods developed for transliteration benchmark tasks such as Named Entity Workshop (NEWS) Shared Task (Li et al., 2009) usually focus on transliterating independent names without properties (or contextual information). For example, "*Kalashnikov*" and "*Calashnikov*" are both acceptable transliterations for "卡拉什尼科夫" (`kǎ lā shí ní kē fū`) in terms of pronunciation. If we know that it refers to a rifle series, however, we should transliterate "卡" to "*Ka*" instead of "*Ca*" here. Therefore, we propose to ground the transliteration results to a KB whenever the contexts are available.

[1]The transliteration systems are publicly available for research purpose at http://nlp.cs.rpi.edu/transliteration/

[2]http://www.colips.org/workshop/news2015/index.html

[3]http://dbpedia.org/resource/

Challenge 2: Information-Losing. As pointed out in (Knight and Graehl, 1998), the information-losing problem of transliteration makes it difficult to invert. For example, “*la*” and “*ra*” are two distinct sounds in English, while they usually collapse to “拉” (lā) in Chinese, which lacks the English “*ra*” sound.

To tackle these two challenges, we propose a novel approach that links a given name to a KB in target language, and subsequently exploits the linking results to correct transliteration hypotheses.

Name String	Transliteration 1	Transliteration 2
雷诺	Renault	Reno
lěi nuò	*French automobile manufacturer*	*city in Nevada*
奥尼尔	(Eugene) O’Neill	(Shaquille) O’Neal
ào ní ěr	*an American playwright*	*an American retired basketball player*
亚瑟	Arthur	Usher
yà sè	*Arthur Pendragon*	*Usher Raymond IV*

Table 1: Chinese-to-English Transliteration Examples for Ambiguous Entities

Challenge 3: Lack of Context. Without specific context, a name string may refer to different entities and thus should be transliterated to different forms. Table 1 shows some instances which require entity disambiguation before transliteration. Take the name “亚瑟” (yà sè) as an example, it has several possible transliteration hypotheses. If we find “雷基”(*Lackey*) in the same document, we can apply collective inference to link their transliteration candidates to a KB. Since “*James Lackey*” appears in the infobox of “*Usher (Singer)*”, we take “*Usher*” and “*Lackey*” as transliterations for “亚瑟” and “雷基” respectively.

Challenge 4: Lack of Training Pairs. Statistical transliteration models usually rely on thousands of name pairs for training. However, it might be costly to collect required training data for low-resource languages. To address this issue, we propose a simple but effective method which transliterates names in a low-resource language using a model trained on one of its similar languages by means of a character mapping table derived from Unicode charts.

2 Approach Overview

Figure 1 illustrates the overall framework of our approach, which consists of four steps as follows.

1. Training. We employ a many-to-many alignment model (m2m-aligner) (Jiampojamarn et al., 2007) to segment and align each transliteration pair in the training data.

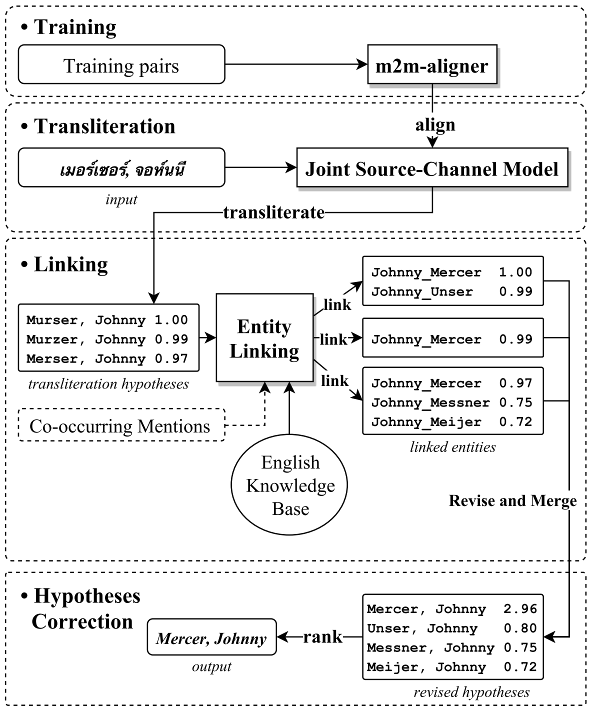

Figure 1: Joint transliteration and linking model framework.

2. Transliteration. For each name in the test set, we apply a joint source-channel model (JSCM) (Li et al., 2004) to generate a list of transliteration hypotheses, where the probabilities of n-grams of transliteration unit pairs are estimated from the alignment result.

3. Linking. We link each transliteration hypothesis to an English KB using a language-independent entity linker (Wang et al., 2015). If context exists, we apply collective inference to link multiple related names simultaneously.

4. Hypotheses Correction. Finally, we revise each hypothesis using the surface forms of the linked entities, and merge and rank the revised hypotheses.

The detailed techniques for each step will be presented in the following sections.

3 Transliteration Hypotheses Generation

We use the joint source-channel model (JSCM) proposed in (Li et al., 2004) as our baseline model.

Given a source name α consisting of a characters $\{x_1, x_2, ..., x_a\}$ and its transliteration β consisting of b characters $\{y_1, y_2, ..., y_b\}$, there exists an alignment γ with K transliteration unit pairs

$\langle s,t\rangle_k = \langle x_i x_{i+1} ... x_{i+p}, y_j y_{j+1} ... y_{j+q}\rangle$, where each s or t corresponds to one or more source or target characters, respectively. The JSCM is an n-gram model defined as

$$\begin{aligned} P(S,T) &= P(\langle s,t\rangle_1, \langle s,t\rangle_2, ..., \langle s,t\rangle_K) \\ &= P(\alpha,\beta,\gamma) \\ &= \prod_{k=1}^{K} P(\langle s,t\rangle_k | \langle s,t\rangle_{k-n+1}^{k-1}) \end{aligned}$$

We can formulate *forward*-transliteration and *back*-transliteration as

$$\bar{\beta} = \arg\max_{\beta,\gamma} P(\alpha,\beta,\gamma)$$

$$\bar{\alpha} = \arg\max_{\alpha,\gamma} P(\alpha,\beta,\gamma)$$

Table 2 shows that the performance of JSCM on *forward*-transliteration (English to foreign language) is comparable with state-of-the-art (Nicolai et al., 2015; Kunchukuttan and Bhattacharyya, 2015) on the NEWS2015 development sets, thereby showing it is a simple but effective model.

Target	**DTL**	**SEQ**	**SMT**	**P**	**M**	**T**	**M+T**	**JSCM**
Hindi	43.5	40.4	36.8	38.8	41.0	37.0	40.5	38.5
Kannada	32.7	35.7	28.1	27.6	32.7	28.9	30.4	26.9
Bengali	37.1	37.8	34.9	35.4	38.2	34.5	36.4	37.1
Tamil	38.5	34.4	29.3	28.6	32.4	31.4	33.4	30.3
Hebrew	61.3	56.6	53.1	54.6	56.4	54.4	54.5	54.9
Thai	36.2	35.8	30.6	-	-	-	-	28.9

Table 2: A comparison of different baseline systems on transliteration accuracy (%). Scores of DTL (DirecTL+), SEQ (Sequitur), and SMT (statistical machine translation) are reported in (Nicolai et al., 2015). Scores of various data representation methods, namely P (character), M (character+boundary marker), T (bigram), and M+T (bigram+boundary marker), are reported in (Kunchukuttan and Bhattacharyya, 2015)

.

In our experiments, we estimate the conditional probability $P(\langle s,t\rangle_k | \langle s,t\rangle_{k-n+1}^{k-1})$ from the alignment result generated by the many-to-many alignment model (m2m-aligner).

Originally designed for letter-to-phoneme conversion, the m2m-aligner has also been used in previous transliteration-related tasks (Jiampojamarn and Kondrak, 2009; Jiampojamarn et al., 2009; Dou et al., 2009; Cook and Stevenson, 2009; Jiampojamarn et al., 2008). We apply the m2m-alingner to the training data to obtain segmentations and alignments. For languages with a large number of characters, training pairs may not cover all characters. As a fallback option, we extend the m2m-aligner's output with pronunciations or romanizations of characters out of the training data. For example, if the Chinese character 孔 (kǒng) is absent in the training set, we use *kong* as its transliteration.

4 Entity Linking

We apply a state-of-the-art language-independent Entity Linker (Wang et al., 2015) to link each transliteration hypothesis to an English KB (DBPedia in our experiment). For each entity name mention m, this entity linker uses the surface form dictionary $\langle f, e_1, e_2, ..., e_k\rangle$, where $e_1, e_2, ..., e_k$ is the set of entities with surface form f in the KB according to their properties (e.g., labels, names, aliases), to locate a list of candidate entities and apply salience ranking by an entropy based approach. After that, it computes similarity scores for each entity mention and candidate entity pair $\langle m, e\rangle$ and re-ranks the candidate entities.

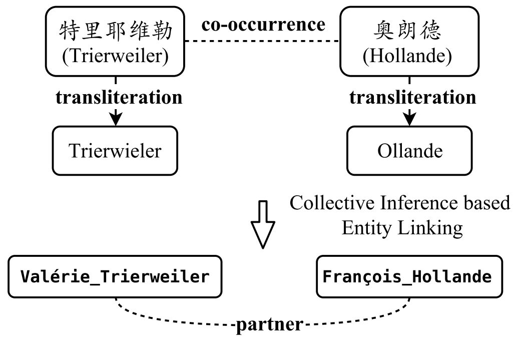

Figure 2: Collective Inference based Entity Linking.

If the context is available, the linker adopts an unsupervised collective inference approach which links multiple entity mentions simultaneously and selects corresponding entity candidates which are most strongly connected in the KB as the final linking results. Figure 2 shows the workflow of Collective Inference based Entity Linking. It first constructs a Mention Context Graph G_m for all entity mentions $M = \{m_1, m_2, ..., m_n\}$ which co-occur within a context window[4] and generates a ranked entity candidate list for each entity mention. KB can be represented as a graph G_k that consists of entities as vertices and weighed relations as edges. Hence, it also constructs a Candidate Graph which

[4]In this paper, we heuristically set the context window to be previous and next three entity mentions.

is a set of graphs $G_c^i (i = 1, 2, ..)$. Here, each graph G_c^i represents a set of entity candidate for mentions in M. Finally, it applies a Candidate Graph collective validation approach that computes similarity scores between G_m and G_c^i and selects G_c^i with the highest score as the final linking results.

In our experiment, we first transliterate all mentions with the JSCM. During the entity linking step, transliteration hypotheses of mentions within a context windows are linked simultaneously.

5 Hypothesis Correction

With a linked entity set $\mathcal{E}_i$ for each transliteration hypothesis β_i, we revise β_i using the entity surface form based on following rules: (1) Split a transliteration hypothesis and the surface form of a linked entity into tokens. (2) Compute string similarity between every hypothesis token and entity token. (3) Revise each hypothesis token to the its most similar entity token.

For example, the top-1 transliteration hypothesis of "พอร์เตอร์, แคเทอรีน แอนน์" (Thai, *Poter, Katherine Anne*) is "*Porter, Catherine Ann*", and `Katherine_Anne_Porter` is one of the linked entities. `Katherine`, `Anne`, and `Porter` are used to revise *Catherine*, *Ann*, and *Porter*, respectively, regardless of the token order.

We compute the score of the revised hypothesis of transliteration β_i and entity candidate c_j as a product of the transliteration score and the linking score.

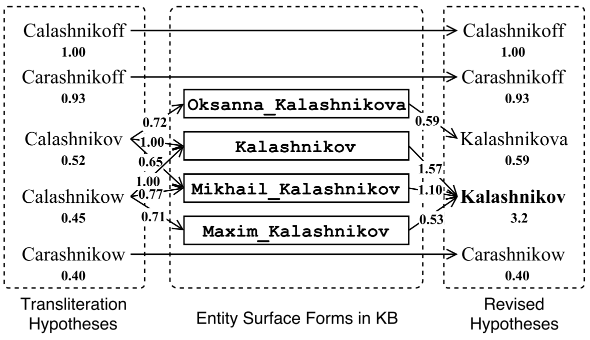

Figure 3: Hypothesis correction.

Figure 3 shows the top-5 transliteration hypotheses of "卡拉什尼科夫" (`kǎ lā shí ní kē fū`, *Kalashnikov*), surface forms of top-3 linked entities for each linkable hypothesis, and the revised hypotheses. Identical revision results from divergent transliteration hypotheses and entity candidates are considered as the same hypothesis, and their scores are summed. After merging revised hypotheses, we rank them and select the top-1 as the final transliteration, which in this example is *Kalashnikov* (3.2).

Ambiguous names which refer to more than one entity may be transliterated in different ways. If the context is provided, we can eliminate ambiguity based on collective inference. For example, in the sentence, "在拉斯维加斯及亨德森之后, 雷诺是内华达州人口第三多的城市 (Reno is the third most populous city in the state of Nevada after Las Vegas and Henderson)," the transliterations of "雷诺" (`léi nuò`) include *Reno*, a city in Nevada, *Renault*, a French automobile manufacturer, and *Raynor*, a virtual role in StarCraft. In order to link it to the correct entity, we first independently transliterate all mentions, namely "拉斯维加斯" (*Las Vegas*), "亨德森" (*Henderson*), "雷诺" (*Reno*) and "内华达" (*Nevada*). Then we apply collective entity linking to these mentions. Since *Reno* has explicit and strong relations with *Las Vegas* (another city in Nevada), *Henderson* (another city in Nevada) and *Nevada* (the state of Reno) in the KB, it is ranked higher than other candidates by the linker.

6 Cross-lingual Projection

For low-resource languages, it is not feasible to directly apply this framework because manually collecting transliteration training pairs takes considerable time and effort.

We observed that some related languages share the same or similar character sets, linguistic characteristics and transliteration conventions. For example, a *virama* is employed to suppress the inherent vowel, namely *schwa*, of its preceding consonant in many Indic scripts. In the light of this fact, it is possible to transfer words or transliteration rules across related languages and thereby avoid collecting extra training data for each language. Therefore, we propose a Unicode name-based projection scheme that transfers IL words to their character equivalents in a high-resource related language so that we can apply the transliteration model trained for the high-resource language. This method is very similar to our recent work on building grapheme-to-phoneme models across related scripts (Deri and Knight, 2016).

In Unicode character code charts[5], most vowels, consonants and signs are assigned a name with the following format:

[5]http://unicode.org/charts

```
SCRIPT TYPE NAME
```

For example, Bengali independent vowel "অ", dependent vowel sign "ু" and consonant "ক" are named `BENGALI LETTER A`, `BENGALI VOWEL SIGN U` and `BENGALI LETTER KA`, respectively.

Utilizing these Unicode character names as a bridge, our approach consists of the following steps: (1) For a low resource language L, select its related language L' whose transliteration pairs can be extracted from existing resources with minimal effort; (2) Construct a L to L' character mapping table based on Unicode character names; (3) Convert a L name α to α' using its corresponding L' characters in the mapping table; (4) Transliterate α' with the model trained on pairs in L'.

To illustrate this idea, the following example shows how to transliterate Hindi word "अमेरिका" (*America*) using a Bengali-to-English transliteration model.

Hindi	Name	Bengali	Name
अ	a	অ	a
क	ka	ক	ka
म	ma	ম	ma
र	ra	র	ra
ा	aa	া	aa
ि	i	ি	i
े	e	ে	e
्	virama	্	virama

Table 3: Hindi to Bengali Mapping Table (Part)

First, we derive a mapping table from Unicode charts of Hindi and Bengali scripts as the the Table 3 shows. For example, the Hindi vowel आ (`DEVANAGARI LETTER AA`) is mapped to its Bengali counterpart ই (`BENGALI LETTER AA`).

Next, the Hindi word "अमेरिका" is converted to "অমরেকিা" character by character following the mapping table. Note that the result is not exactly the same as "আমরেকিা", the actual Bengali word representing "*America*". Finally, "অমরেকিা" is transliterated into "*America*" using the Bengali-to-English transliteration model.

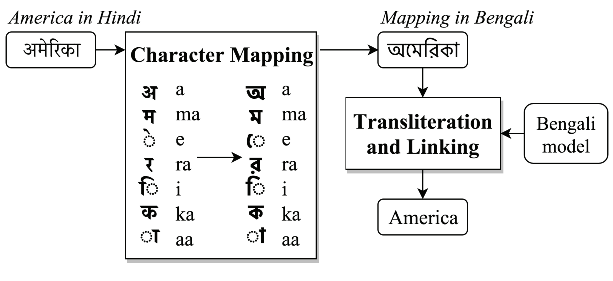

Figure 4: Transliterate Hindi using Bengali model.

7 Experiments

In this section we will present experimental results for context-independent, context-dependent, and cross-lingual projection settings respectively.

7.1 Context-Independent Transliteration

We train transliteration models of six languages with the NEWS2015 data sets and use corresponding *development* sets as our test set since the official test sets are not publicly available. Additionally, because some source names lack matched entities in DBpedia, we also evaluate with subsets containing only linkable names, whose gold transliterations match at least one entity in DBpedia. Table 4 summarizes the data statistics[6,7]. We evaluate the performance based on a strict *accuracy* metric by checking whether our top 1 hypothesis of each name exactly matches the ground-truth transliteration.

Source	Train	Test	Linkable
Hindi	11,946	997	606
Kannada	9,955	1,000	571
Bengali	13,855	986	574
Tamil	9,959	1,000	537
Hebrew	9,501	1,000	924
Thai	25,597	1,994	1,478

Table 4: # of name pairs in NEWS2015 data sets.

Source	Overall			Linkable only	
	JSCM	+EL	+LM	JSCM	+EL
Hindi	40.3	**44.8**	41.2	42.7	54.3
Kannada	29.8	**37.6**	34.2	30.0	41.5
Bengali	49.4	**52.0**	49.5	45.1	54.9
Tamil	20.2	**29.0**	24.6	23.5	41.5
Hebrew	21.5	**37.2**	27.5	21.7	38.0
Thai	29.3	**44.0**	32.8	29.3	44.8

Table 5: *Back*-transliteration accuracy on NEWS2015 development sets (%).

We use JSCM as our baseline model, which has comparable performance with other models used in the NEWS Shared Task. Since we focus on *back*-transliteration from low-resource languages to English in this work, and only Thai has the *back*-transliteration task and data set, we reverse the source and target names of other languages in our experiments. In addition, we train a unigram language model (LM) from Gigaword[8] to revise

[6]MSR India owns the English-Hindi, English-Tamil, English-Kannada, English-Bengali and English-Hebrew task corpora. http://research.microsoft.com/india

[7]NECTEC owns the Thai-English task corpus.

[8]https://catalog.ldc.upenn.edu/LDC2003T05

Source	Name	Top-1 Hypothesis	+EL	Comment
Hindi	गैरी	Garr**i**	Garr**y**	vowel
	वोर्पोमेर्न	Vorpo**m**ern	Vorpo**mm**ern	double consonants
Kannada	ಮಾಂಜ್ರೆಕರ್	Man**z**rekar	Man**j**rekar	consonant
	ಬುಚಾರೆಸ್ಟ್	Bu**tch**arest	Bu**ch**arest	consonant
Bengali	ক্যাম্পারডাউন	Camp**a**rdown	Camp**e**rdown	vowel
	মুসোলিনী	Mu**s**olini	Mu**ss**olini	double consonants
Tamil	ஜயெதி	Jaya**d**i	Jaya**t**i	consonant
	கல்யூக்	Kalyu**k**	Kalyu**g**	consonant
Hebrew	קאלינינגראד	**C**aliningrad	**K**aliningrad	consonant
	שישילוב	Shish**i**lov	Shish**e**lov	vowel
Thai	สกาลาแวก	Scalawa**gue**	Scalawa**g**	consonant
	เฟอร์ลิงเกตติ	Ferlin**k**eti	Ferlin**gh**etti	consonant

Table 6: Transliteration correction examples.

transliteration hypotheses and compare its performance with the entity linker. Gigaword contains 4.16 billion words, 267 million named entities and 7.4 million unique named entities.

The overall performance is shown in Table 5. We can see that entity linking improves the transliteration accuracy for all languages, especially for the linkable subsets. Previous results on *back*-transliteration are only available for Thai-to-English, and our top-1 accuracy (44.0%) notably advances the previous highest score (39.5%) reported in (Nicolai et al., 2015). Besides, our experiments show that the entity linker outperforms the language model trained on a large corpus.

In Table 6, we list some correction examples for languages we evaluate. Entity Linking has mainly made three types of corrections as follows.

1. Double consonants. Repeated consonant letters in English, such as "*tt*", are usually transliterated into a single character in languages not written in Roman script. Since double consonants are less frequent than single ones, statistical models tend to *back*-transliterate a character into a single corresponding English letter, e.g., "斯"(sī) to "*s*" instead of "*ss*".

2. Consonant. Because some orthographies lack characters to represent all English consonants, different English consonants may correspond to the same character in other languages. For example, "*k*" and "*g*" are usually transliterated into "க" in Tamil.

3. Vowel. The correspondences between English vowel letters and phonemes are complex. A vowel letter may be pronounced in different ways, e.g., the three *a*'s in ***banana***, while distinct vowel letters may have the same pronunciation, e.g., the first *e* and second *i* in ***ingredient***. Such inconsistency makes *back*-transliteration from shallow orthographies more difficult.

Although entity linking explicitly improves the transliteration quality, we observe that some correct transliteration hypotheses are mistakenly revised due to the lack of context and corresponding entities. For example, "लखपति" is correctly transliterated to *Lakhpati* but subsequently revised to *Lakhpat* because *Lakhpati* is absent in the KB.

7.2 Context-Dependent Transliteration

In order to evaluate context-dependent transliteration, we train a Chinese-to-English transliteration model on 44,146 name pairs (Ji et al., 2009), and use the Chinese-to-English Entity Discovery and Linking data set in NIST TAC-KBP2015 evaluation (Ji et al., 2015) since the NEWS2015 Chinese data set only contains transliteration pairs without any contextual information. This data set contains 160 documents and 11,066 Chinese mentions, including 1,239 person names (469 unique ones). Each name has a ground-truth transliteration derived from English KB title.

JSCM	Non-Collective	Collective
32.9	56.4	57.1

Table 7: Impact of collective inference on transliteration (%).

Table 7 presents the results for person names. We can see that by exploiting the contextual information, collective inference provides more accurate entity linking and hence further enhances transliteration.

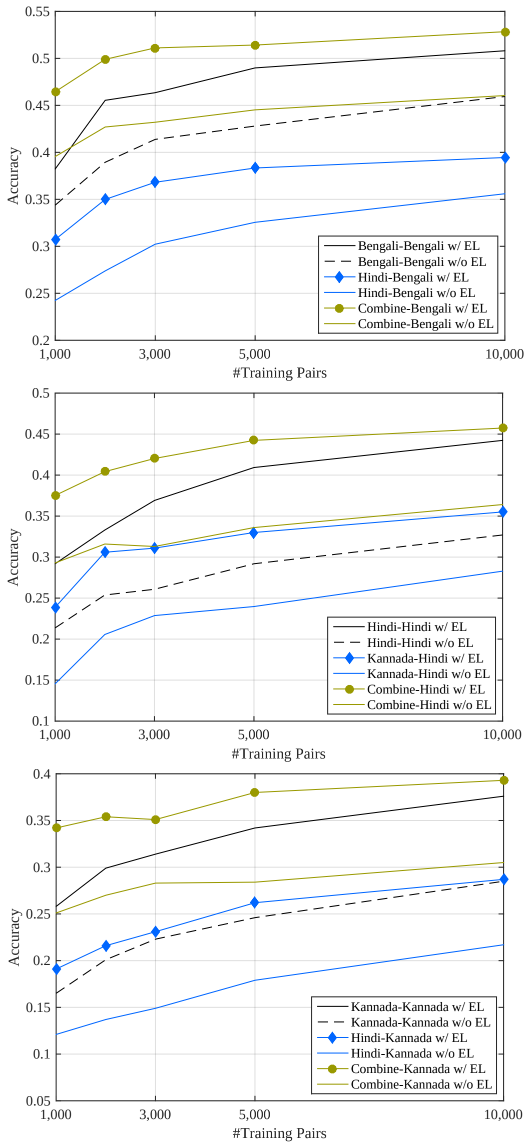

Figure 5: Transliteration accuracy score with different sizes of training pairs. The first language of each pair is the language that the model is trained on, and the second one is the test language.

7.3 Cross-lingual Projection

In projection experiments, we train transliteration models on the NEWS2015 data sets described in Section 7.1, and evaluate each with other languages using the projection approach proposed in Section 6. Among the six languages, our results show that the Hindi model is the best for transliterating Bengali and Kannada names, while the Kannada model is the best for transliterating Hindi names. Learning curves of these three pairs depicted in Figure 5 show that it is feasible to transliterate names using a model trained on a related language without extra data. For example, a training set of $10,000$ Hindi-English pairs achieves the same performance as $4,000$ Kannada-English pairs on transliterating Kannada names.

Nevertheless, we still observe a performance gap between training from the related language and the source language itself. This indicates that after applying our character mapping-based method, dissimilarities may still exist between the representations of the same name from two related languages. For example, because "भ" is the most similar Hindi character for Bengali character "ভ", we mapped "ভ" to "भ". However, apart from their common transliteration "*bh*", "ভ" can also be transliterated to "*v*", such as "ভেঙ্কটেশ" (*Venkatesh*), thereby leading to some incorrect transliterations, e.g., "अभनिभास" (*Abhinbhaas*) to "*Abhinvass*". Another example is that the combination of "ॉ" and "क" is usually transliterated to "*ock*" in the Bengali training set, whereas its Hindi equivalent "ে াক" is usually transliterated to "*och*" or "*ok*", and therefore "लॉकेट"(*locket*) is incorrectly transliterated to "*loket*". In addition, representations referring to the same entity may have divergent origins in different languages (e.g., "*German*" has variants including "*Germanisch*", "*Deutsch*", "*Alemannisch*", "*niemy*" and "*Sachsen*").

However, we see that such performance gap can be narrowed or filled by entity linking. In Figure 5, *Kannada-Hindi w/EL* and *Hindi-Kannada w/EL* even outperform *Hindi-Hindi w/o EL* and *Kannada-Kannada w/o EL*, respectively.

We also train combination models using different sizes of pairs of a source language and $10,000$ pairs of its related language. Such a combination dramatically improves the performance, which means that by borrowing pairs from a related language, we can develop a high-performance model with only a small number of transliteration pairs of the source language.

We also find that it is difficult to construct mapping tables for Thai and Hebrew since they share few similar character names with Hindi, Kannada, Bengali, and Tamil. Additionally, despite of the fact that Kannada and Tamil lie on the same language family branch, they have evolved independently for centuries and have different representations of sounds, which to some extent explains why the projection between them is not as effective. For example, in Kannnda script, there are different letters for `ka` (ಕ) and `ga` (ಗ), whereas Tamil only uses one letter க (ka).

7.4 Remaining Challenges

Regardless of our improvement and promising results, the overall strict accuracy name transliteration is still quite low. We categorize the remaining challenges as follows.

- **Name Segmentation**. Some additional splitting or merging operations are needed for some names. For example, "अग्निपुराण" in Hindi should be transliterated into two tokens "*Agni Purana*".
- **Source Language Specific Features**. For example, "*-istan*" is a common country suffix in Turkish and Persian, and thus it can be ignored during transliteration (e.g., when transliterating "*Gürcistan*" in Turkish to English, we can focus on transliterating "*Gürc*" to "*Georgia*").
- **Entity Profile**. Name transliteration might follow specific conventions based on the entity's origin, gender, title and characteristic. For example, "*Monroe*" is transliterated to "门罗" (mén luó) in "*James Monroe*", the fifth President of the United States, while "梦露" (mèng lù) in "*Marilyn Monroe*", a famous American actress, where "门", "罗", "梦" and "露" refer to "*door*", "*net*", "*dream*" and "*dew*", respectively. For celebrities with corresponding entities in the KB, the collective inference method we employ can resolve the ambiguity and hence generate correct transliterations, while it does not work for out-of-KB ones. In order to transliterate such out-of-KB names, some of their properties, such as gender, need to be inferred from the text.

8 Related Work

In terms of transliteration unit, existing machine transliteration models can be classified into three categories, phoneme-based (Knight and Graehl, 1997; Lee and Choi, 1998; Wan and Verspoor, 1998; Jung et al., 2000; Meng et al., 2001; Oh and Choi, 2002; Virga and Khudanpur, 2003; Gao et al., 2005), grapheme-based (Li et al., 2004; Zhang et al., 2004; Ekbal et al., 2006; Ganesh et al., 2008; Das et al., 2009; Chinnakotla et al., 2010; Finch and Sumita, 2010), and hybrid (Al-Onaizan and Knight, 2002; Bilac and Tanaka, 2004; Oh and Choi, 2005; Oh et al., 2006; Kim et al., 1999).

Since names are inherently associated with entities, it is natural to leverage entity linking to improve name transliteration. To the best of our knowledge, this is the first study using entity linking results to revise transliteration hypotheses. We also take the specific context of a name into consideration to improve the quality of entity linking and reduce ambiguity.

Additionally, to tackle the data sparsity challenge in low-resource languages, we propose a simple but effective cross-lingual projection approach to take advantage of resources in related languages. Similar cross-lingual projection methods based on data/annotation transfer have also been exploited for other Natural Language Processing tasks, including relation extraction, data annotation, entity recognition, and grapheme-to-phoneme models (Xia and Lewis, 2007; Padó and Lapata, 2009; Kim et al., 2010; Faruqui and Kumar, 2015; Deri and Knight, 2016).

9 Conclusions and Future Work

For many names we need to know the real-world entities they refer to before generating their correct transliterations. In this paper we developed a novel context-aware name transliteration approach by leveraging Entity Linking and related language projection. Experiments have demonstrated that our approach can significantly enhance the transliteration performance. In the future we will explore more knowledge from the KB such as types and properties of entities to improve disambiguation and transliteration. We will also aim to incorporate morphology analysis, acquire and incorporate language-specific and culture-specific characteristics to address the remaining challenges.

Acknowledgment

The work was supported by the U.S. DARPA LORELEI Program No. HR0011-15-C-0115 and ARL/ARO MURI W911NF-10-1-0533. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In *ACL2002*.

Slaven Bilac and Hozumi Tanaka. 2004. A hybrid back-transliteration system for Japanese. In *COLING2004*.

Manoj K Chinnakotla, Om P Damani, and Avijit Satoskar. 2010. Transliteration for resource-scarce languages. *ACM Transactions on Asian Language Information Processing*.

Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message normalization. In *ACL2009*.

Amitava Das, Asif Ekbal, Tapabrata Mandal, and Sivaji Bandyopadhyay. 2009. English to Hindi machine transliteration system at NEWS 2009. In *ACL2009*.

Aliya Deri and Kevin Knight. 2016. Grapheme-to-phoneme models for (almost) any language. In *ACL2016*.

Qing Dou, Shane Bergsma, Sittichai Jiampojamarn, and Grzegorz Kondrak. 2009. A ranking approach to stress prediction for letter-to-phoneme conversion. In *ACL2009*.

Asif Ekbal, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. 2006. A modified joint source-channel model for transliteration. In *ACL2006*.

Manaal Faruqui and Shankar Kumar. 2015. Multilingual open relation extraction using cross-lingual projection. *NAACL2015*.

Andrew M Finch and Eiichiro Sumita. 2010. A Bayesian model of bilingual segmentation for transliteration. In *IWSLT2010*.

Surya Ganesh, Sree Harsha, Prasad Pingali, and Vasudeva Verma. 2008. Statistical transliteration for cross language information retrieval using HMM alignment model and CRF. In *IJCNLP2008*.

Wei Gao, Kam-Fai Wong, and Wai Lam. 2005. Phoneme-based transliteration of foreign names for OOV problem. In *IJCNLP2005*.

Heng Ji, Ralph Grishman, Dayne Freitag, Matthias Blume, John Wang, Shahram Khadivi, Richard Zens, and Hermann Ney. 2009. Name extraction and translation for distillation. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of TAC-KBP2015 tri-lingual entity discovery and linking. In *TAC2015*.

Sittichai Jiampojamarn and Grzegorz Kondrak. 2009. Online discriminative training for grapheme-to-phoneme conversion. In *ISCA2009*.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *NAACL2007*.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2008. Joint processing and discriminative training for letter-to-phoneme conversion. In *ACL2008*.

Sittichai Jiampojamarn, Aditya Bhargava, Qing Dou, Kenneth Dwyer, and Grzegorz Kondrak. 2009. DirecTL: a language-independent approach to transliteration. In *ACL2009*.

Sung Young Jung, SungLim Hong, and Eunok Paek. 2000. An English to Korean transliteration model of extended Markov window. In *ACL2000*.

Jung-Jae Kim, Jae-Sung Lee, and Key-Sun Choi. 1999. Pronunciation unit based automatic English-Korean transliteration model using neural network. In *Proceedings of Korea Cognitive Science Association*.

Seokhwan Kim, Minwoo Jeong, Jonghoon Lee, and Gary Geunbae Lee. 2010. A cross-lingual annotation projection approach for relation detection. In *COLING2010*.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *EACL1997*.

Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2015. Data representation methods and use of mined corpora for Indian language transliteration. In *ACL2015*.

Jae-Sung Lee and Key-Sun Choi. 1998. English to Korean statistical transliteration for information retrieval.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *ACL2004*.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009. Report of NEWS 2009 machine transliteration shared task. In *ACL2009*.

Helen M Meng, Wai-Kit Lo, Berlin Chen, and Karen Tang. 2001. Generating phonetic cognates to handle named entities in English-Chinese cross-language spoken document retrieval. In *ASRU2001*.

Garrett Nicolai, Bradley Hauer, Mohammad Salameh, Adam St Arnaud, Ying Xu, Lei Yao, and Grzegorz Kondrak. 2015. Multiple system combination for transliteration. In *ACL2015*.

Jong-Hoon Oh and Key-Sun Choi. 2002. An English-Korean transliteration model using pronunciation and contextual rules. In *COLING2002*.

Jong-Hoon Oh and Key-Sun Choi. 2005. An ensemble of grapheme and phoneme for machine transliteration. In *IJCNLP2005*.

Jong-Hoon Oh and Hitoshi Isahara. 2007. Machine transliteration using multiple transliteration engines and hypothesis re-ranking. *MT Summit*.

Jong-Hoon Oh, Key-Sun Choi, and Hitoshi Isahara. 2006. A machine transliteration model based on correspondence between graphemes and phonemes. *ACM Transactions on Asian Language Information Processing*.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *JAIR*.

Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of proper names in cross-lingual information retrieval. In *ACL2003*.

Stephen Wan and Cornelia Maria Verspoor. 1998. Automatic English-Chinese name transliteration for development of multilingual resources. In *COLING1998*.

Han Wang, Jin Guang Zheng, Xiaogang Ma, Peter Fox, and Heng Ji. 2015. Language and domain independent entity linking with quantified collective validation. In *EMNLP2015*.

Fei Xia and William D Lewis. 2007. Multilingual structural projection across interlinear text. In *NAACL2007*.

Min Zhang, Haizhou Li, and Jian Su. 2004. Direct orthographical mapping for machine transliteration. In *COLING2004*.