Everybody loves a rich cousin: An empirical study of transliteration through bridge languages

Mitesh M. Khapra
Indian Institute of Technology Bombay,
Powai, Mumbai 400076,
India
miteshk@cse.iitb.ac.in

A Kumaran
Microsoft Research India,
Bangalore,
India
a.kumaran@microsoft.com

Pushpak Bhattacharyya
Indian Institute of Technology Bombay,
Powai, Mumbai 400076,
India
pb@cse.iitb.ac.in

Abstract

Most state of the art approaches for machine transliteration are data driven and require significant parallel names corpora between languages. As a result, developing transliteration functionality among n languages could be a resource intensive task requiring parallel names corpora in the order of ${}^{n}C_{2}$. In this paper, we explore ways of reducing this high resource requirement by leveraging the available parallel data between subsets of the n languages, transitively. We propose, and show empirically, that reasonable quality transliteration engines may be developed between two languages, X and Y, even when no direct parallel names data exists between them, but only transitively through language Z. Such systems alleviate the need for $O({}^{n}C_{2})$ corpora, significantly. In addition we show that the performance of such transitive transliteration systems is in par with direct transliteration systems, in practical applications, such as CLIR systems.

1 Introduction

Names and Out Of Vocabulary (OOV) terms appear very frequently in written and spoken text and hence play a very important role in several Natural Language Processing applications. Several studies have shown that handling names correctly across languages can significantly improve the performance of CLIR Systems (Mandl and Womser-Hacker, 2004) and the utility of machine translation systems. The fact that translation lexicons or even statistical dictionaries derived from parallel data do not provide a good coverage of name and OOV translations, underscores the need for good transliteration engines to transform them between the language.

The importance of machine transliteration, in the above context, is well realized by the research community and several approaches have been proposed to solve the problem. However, most state of the art approaches are data driven and require significant parallel names corpora between languages. Such data may not always be available between every pair of languages, thereby limiting our ability to support transliteration functionality between many language pairs, and subsequently information access between languages. For example, let us consider a practical scenario where we have six languages from four different language families as shown in Figure 1. The nodes in the graph represent languages and the edges indicate the availability of data between that language pair and thus the availability of a Machine Transliteration system for that pair. It is easy to see the underlying characteristics of the graph. Data is available between a language pair due to one of the following three reasons:

Politically related languages: Due to the political dominance of English it is easy to obtain parallel names data between English and most languages.

Genealogically related languages: Arabic and Hebrew share a common origin and there is a significant overlap between their phoneme and grapheme inventory. It is easy to obtain parallel names data between these two languages.

Demographically related languages: Hindi and Kannada are languages spoken in the Indian subcontinent, though they are from different language families. However, due to the shared culture and demographics, it is easy to create parallel names data between these two languages.

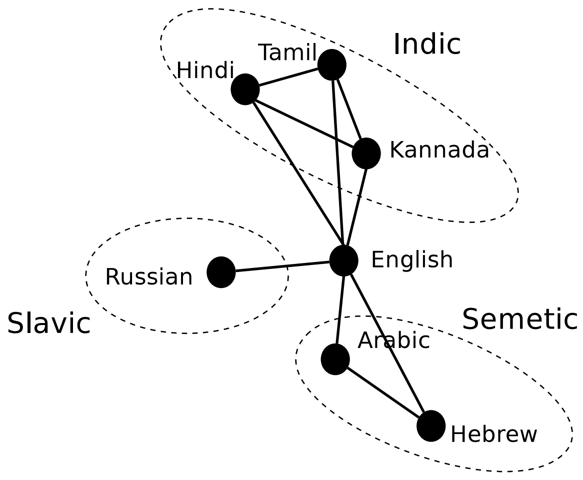


Figure 1: A connected graph of languages

On the other hand, for politically, demographically and genealogically unrelated languages such as, say, Hindi and Hebrew, parallel data is not readily available, either due to the unavailability of skilled bilingual speakers. Even the argument of using Wikipedia resources for such creation of such parallel data does not hold good, as the amount of interlinking may be very small to yield data. For example, only 800 name pairs between Hindi and Hebrew were mined using a state of the art mining algorithm (Udupa et al., 2009), from Wikipedia interwiki links.

We propose a methodology to develop a practical Machine Transliteration system between any two nodes of the above graph, provided a two-step path exists between them. That is, even when no parallel data exists between X & Y but sufficient data exists between X & Z and Z & Y it is still possible to develop transliteration functionality between X & Y by combining a $X \rightarrow Z$ system with a $Z \rightarrow Y$ system. For example, given the graph of Figure 1, we explore the possibility of developing transliteration functionality between Hindi and Russian even though no direct data is available between these two languages. Further, we show that in many cases the bridge language can be suitably selected to ensure optimal MT accuracy.

To establish the practicality and utility of our approach we integrated such a bridge transliteration system with a standard CLIR system and compared its performance with that of a direct transliteration system. We observed that such a bridge system performs well in practice and in specific instances results in improvement in CLIR performance over a baseline system further strengthening our claims that such bridge systems are good practical solutions for alleviating the resource scarcity problem.

To summarize, our main contributions in this paper are:

1. Constructing bridge transliteration systems and establishing empirically their quality.

2. Demonstrating their utility in providing practical transliteration functionality between two languages X & Y with no direct parallel data between them.

3. Demonstrating that in specific cases it is possible to select the bridge language so that optimal Machine Transliteration accuracy is ensured while stepping through the bridge language.

1.1 Organization of the Paper

This paper is organized in the following manner. In section 2 we present the related work and highlight the lack of work on transliteration in resource scarce scenarios. In section 3 we discuss the methodology of bridge transliteration. Section 4 discusses the experiments and datasets used. Section 4.3 discusses the results and error analysis. Section 5 discusses orthographic characteristics to be considered while selecting the bridge language. Section 6 demonstrates the effectiveness of such bridge systems in a practical scenario, *viz.*, Cross Language Information Retrieval. Section 7 concludes the paper, highlighting future research issues.

2 Related Work

Current models for transliteration can be classified as grapheme-based, phoneme-based and hybrid models. Grapheme-based models, such as, Source Channel Model (Lee and Choi, 1998), Maximum Entropy Model (Goto et al., 2003), Conditional Random Fields (Veeravalli et al., 2008) and Decision Trees (Kang and Choi, 2000) treat transliteration as an orthographic process and try to map the source language graphemes directly to the target language graphemes. Phoneme based models, such as, the ones based on Weighted Finite State

Transducers (WFST) (Knight and Graehl, 1997) and extended Markov window (Jung et al., 2000) treat transliteration as a phonetic process rather than an orthographic process. Under such frameworks, transliteration is treated as a conversion from source grapheme to source phoneme followed by a conversion from source phoneme to target grapheme. Hybrid models either use a combination of a grapheme based model and a phoneme based model (Stalls and Knight, 1998) or capture the correspondence between source graphemes and source phonemes to produce target language graphemes (Oh and Choi, 2002).

A significant shortcoming of all the previous works was that none of them addressed the issue of performing transliteration in a resource scarce scenario, as there was always an implicit assumption of availability of data between a pair of languages. In particular, none of the above approaches address the problem of developing transliteration functionality between a pair of languages when no direct data exists between them but sufficient data is available between each of these languages and an intermediate language. Some work on similar lines has been done in Machine Translation (Wu and Wang, 2007) wherein an intermediate bridge language (say, Z) is used to fill the data void that exists between a given language pair (say, X and Y). In fact, recently it has been shown that the accuracy of a $X \rightarrow Z$ Machine Translation system can be improved by using additional $X \rightarrow Y$ data provided Z and Y share some common vocabulary and cognates (Nakov and Ng, 2009). However, no such effort has been made in the area of Machine Transliteration. To the best of our knowledge, this work is the first attempt at providing a practical solution to the problem of transliteration in the face of resource scarcity.

3 Bridge Transliteration Systems

In this section, we explore the salient question *"Is it possible to develop a practical machine transliteration system between X and Y, by composing two intermediate $X \rightarrow Z$ and $Z \rightarrow Y$ transliteration systems?"* We use a standard transliteration methodology based on orthography for all experiments (as outlined in section 3.1), to ensure the applicability of the methodology to a variety of languages.

3.1 CRF based transliteration engine

Conditional Random Fields ((Lafferty et al., 2001)) are undirected graphical models used for labeling sequential data. Under this model, the conditional probability distribution of the target word given the source word is given by,

$$P(Y|X;\lambda) = \frac{1}{N(X)} \cdot e^{\sum_{t=1}^{T}\sum_{k=1}^{K}\lambda_k f_k(Y_{t-1},Y_t,X,t)} \tag{1}$$

where,

$$\begin{aligned} X &= source\ word \\ Y &= target\ word \\ T &= length\ of\ source\ word \\ K &= number\ of\ features \\ \lambda_k &= feature\ weight \\ N(X) &= normalization\ constant \end{aligned}$$

CRF++ [1], an open source implementation of CRF was used for training and decoding (*i.e.* transliterating the names). GIZA++ (Och and Ney, 2003), a freely available implementation of the IBM alignment models (Brown et al., 1993) was used to get character level alignments for the name pairs in the parallel names training corpora. Under this alignment, each character in the source word is aligned to zero or more characters in the corresponding target word. The following features are then generated using this character-aligned data (here e_i and h_i form the *i*-th pair of aligned characters in the source word and target word respectively):

- h_i and e_j such that $i-2 \leq j \leq i+2$
- h_i and source character bigrams ($\{e_{i-1}, e_i\}$ or $\{e_i, e_{i+1}\}$)
- h_i and source character trigrams ($\{e_{i-2}, e_{i-1}, e_i\}$ or $\{e_{i-1}, e_i, e_{i+1}\}$ or $\{e_i, e_{i+1}, e_{i+2}\}$)
- h_i, h_{i-1} and e_j such that $i-2 \leq j \leq i+2$
- h_i, h_{i-1} and source character bigrams
- h_i, h_{i-1} and source character trigrams

[1] http://crfpp.sourceforge.net/

3.2 Bridge Transliteration Methodology

In this section, we outline our methodology for composing transitive transliteration systems between X and Y, using a bridge language Z, by chaining individual direct transliteration systems. Our approach of using bridge transliteration for finding the best target string (Y^*), given the input string X can be represented by the following probabilistic expression:

$$\begin{aligned} Y^* &= \arg\max_Y P(Y|X) \\ &= \sum_Z P(Y,Z|X) \\ &= \sum_Z P(Y|Z,X) * P(Z|X) \end{aligned} \tag{2}$$

We simplify the above expression, by assuming that Y is independent of X given Z; the linguistic intuition behind this assumption is that the *top-k* outputs of the $X \rightarrow Z$ system corresponding to a string in X, capture all the transliteration information necessary for transliterating to Y. Subsequently, in section 5 we discuss the characteristics of the effective bridge languages to maximize the capture of necessary information for the second stage of the transliteration, namely for generating correct strings of Z. Thus,

$$Y^* = \sum_Z P(Y|Z) * P(Z|X) \tag{3}$$

The probabilities $P(Y|Z)$ and $P(Z|X)$ in Equation (3) are derived from the two stages of the bridge system. Specifically, we assume that the parallel names corpora are available between the language pair, X and Z, and the language pair, Z and Y. We train two baseline CRF based transliteration systems (as outlined in Section 3.1), between the language X and Z, and Z and Y. Each name in language X was provided as an input into $X \rightarrow Z$ transliteration system, and the top-10 candidate strings in language Z produced by this first stage system were given as an input into the second stage system $Z \rightarrow Y$. The results were merged using Equation (2). Finally, the top-10 outputs of this system were selected as the output of the bridge system.

4 Experiments

It is a well known fact that transliteration is lossy, and hence the transitive systems may be expected to suffer from the accumulation of errors in each stage, resulting in a system that is of much poorer quality than a direct transliteration system. In this section, we set out to quantify this expected loss in accuracy, by a series of experiments in a set of languages using bridge transliteration systems and a baseline direct systems. We conducted a comprehensive set of experiments in a diverse set of languages, as shown in Figure 1, that include English, Indic (Hindi and Kannada), Slavic (Russian) and Semitic (Arabic and Hebrew) languages. The datasets and results are described in the following subsections.

4.1 Datasets

To be consistent, for training each of these systems, we used approximately 15K name pairs corpora (as this was the maximum data available for some language pairs). While we used the NEWS 2009 training corpus (Li et al., 2009) as a part of our training data, we enhanced the data set to about 15K by adding more data of similar characteristics (such as, name origin, domain, length of the name strings, *etc.*), taken from the same source as the original NEWS 2009 data. For languages such as Arabic and Hebrew which were not part of the NEWS 2009 shared task, the data was created along the same lines. All results are reported on the standard NEWS 2009 test set, wherever applicable. The test set consists of about 1,000 name pairs in languages X and Y; to avoid any bias, it was made sure that there is no overlap between the test set with the training sets of both the $X \rightarrow Z$ and $Z \rightarrow Y$ systems. To establish a baseline, the same CRF based transliteration system (outlined in Section 3.1) was trained with a 15K name pairs corpora between the languages $X \rightarrow Y$. The same test set used for testing the transitive systems was used for testing the direct system as well. As before, to avoid any bias, we made sure that there is no overlap between the test set and the training set for the direct system as well.

4.2 Results

We produce top-10 outputs from the bridge system as well from the direct system and compare their performance. The performance is measured using the following standard measures, *viz.*, top-1 accuracy (ACC-1) and Mean F-score. These measures are described in detail in (Li et al., 2009). Table 1

Language Pair	ACC-1	Relative change in ACC-1	Mean F-score	Relative change in Mean F-score
Hin-Rus	0.507		0.903	
Hin-Eng-Rus	0.466	-8.08%	0.886	-1.88%
Hin-Ara	0.458		0.897	
Hin-Eng-Ara	0.420	-8.29%	0.876	-2.34%
Eng-Heb	0.544		0.917	
Eng-Ara-Heb	0.544	0%	0.917	0%
Hin-Eng	0.422		0.884	
Hin-Kan-Eng	0.382	-9.51%	0.871	-1.47%

Table 1: Stepping through an intermediate language

presents the performance measures, both for a direct system (say, Hin-Rus), and a transitional system (say, Hin-Eng-Rus), in 4 different transitional systems, between English, Indic, Semitic and Slavic languages. In each case, we observe that the transitional systems have a slightly lower quality, with an absolute drop in accuracy (ACC-1) of less than 0.05 (relative drop under 10%), and an absolute drop in Mean F-Score of 0.02 (relative drop under 3%).

4.3 Analysis of Results

Intuitively, one would expect that the errors of the two stages of the transitive transliteration system (i.e., $X \rightarrow Z$, and $Z \rightarrow Y$) to compound, leading to a considerable loss in the overall performance of the system. Given that the accuracies of the direct transliteration systems are as given in Table 2, the transitive systems are expected to have accuracies close to the product of the accuracies of the individual stages, for independent systems.

Language Pair	ACC-1	Mean F-Score
Hin-Eng	0.422	0.884
Eng-Rus	0.672	0.935
Eng-Ara	0.514	0.905
Ara-Heb	1.000	1.000
Hin-Kan	0.433	0.879
Kan-Eng	0.434	0.886

Table 2: Performance of Direct Transliteration Systems

However, as we observe in Table 1, the relative drop in the accuracy (ACC-1) is less than 10% from that of the direct system, which goes against our intuition. To identify the reasons for the better than expected performance, we performed a detailed error analysis of each stage of the bridge transliteration systems, and the results are reported in Tables 3 – 5. We draw attention to two interesting facts which account for the better than expected performance of the bridge system:

Improved 2nd stage performance on correct inputs: In each one of the cases, as expected, the ACC-1 of the first stage is same as the ACC-1 of the $X \rightarrow Z$ system. However, we notice that the ACC-1 of the second stage *on the correct strings output in the first stage*, is significantly better than the the ACC-1 of the $Z \rightarrow Y$ system! For example, the ACC-1 of the Eng-Rus system is 67.2% (see Table 2), but, that of the 2nd stage Eng-Rus system is 77.8%, namely, on the strings that are transliterated correctly by the first stage. Our analysis indicate that there are two reasons for such improvement: First, the strings that get transliterated correctly in the first stage are typically shorter or less ambiguous and hence have a better probability of correct transliterations in the both stages. This phenomenon could be verified empirically: Names like गोपाल {Gopal}, रमेश {Ramesh}, राम {Ram} are shorter and in general have less ambiguity on target orthography. Second, also significantly, the use of top-10 outputs from the first stage as input to the second stage provides a better opportunity for the second stage to produce correct string in Z. Again, this phenomenon is verified by providing increasing number of *top-n* results to the 2nd stage.

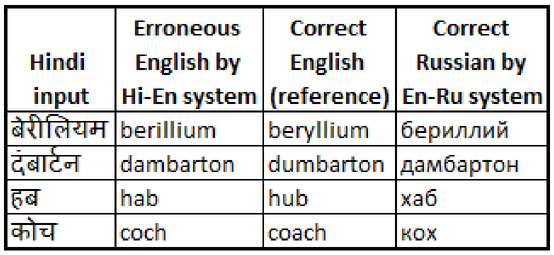
Hi→En→Ru		En → Ru (Stage-2)		Stage-2 Acc.
		Correct	**Error**	
Hi→En (Stage-1)	**Correct**	263	75	**77.81%**
	Error	**119**	362	**24.74%**

Table 3: Error Analysis for Hi→En→Ru

Hi→En→Ar		En → Ar (Stage-2)		Stage-2 Acc.
		Correct	**Error**	
Hi→En (Stage-1)	**Correct**	221	127	**63.50%**
	Error	**119**	340	**25.70%**

Table 4: Error Analysis for Hi→En→Ar

2nd stage error correction on incorrect inputs: The last rows in each of the above tables 3 – 5 report the performance of the second stage system on strings that were transliterated incorrectly by the first stage. While we expected the second row to produce incorrect transliterations nearly for all inputs (as the input themselves were incorrect in Z), we find to our surprise that upto 25% of the erroneous strings in Z were getting transliterated correctly in Y! This provides credence to our hypothesis that sufficient transliteration information is captured in the 1st stage output (*even when incorrect*) that may be exploited in the 2nd stage. Empirically, we verified that in most cases (nearly 60%) the errors were due to the incorrectly transliterated vowels, and in many cases, they get corrected in the second stage, and re-ranked higher in the output. Figure 2 shows a few examples of such error corrections in the second stage.



Hindi input	Erroneous English by Hi-En system	Correct English (reference)	Correct Russian by En-Ru system
बेरीलियम	berillium	beryllium	бериллий
दंबार्टन	dambarton	dumbarton	дамбартон
हब	hab	hub	хаб
कोच	coch	coach	кох

Figure 2: Examples of error corrections

Hi→Ka→En		Ka → En (Stage-2)		Stage-2 Acc.
		Correct	**Error**	
Hi→Ka (Stage-1)	**Correct**	225	196	**53.44%**
	Error	**151**	400	**27.40%**

Table 5: Error Analysis for Hi→Ka→En

5 Characteristics of the bridge language

An interesting question that we explore in this section is *"how the choice of bridge language influence the performance of the bridge system?"*. The underlying assumption in transitive transliteration systems (as expressed in Equation 3), is that *"Y is independent of X given Z"*. In other words, we assume that the representations in the language will Z *"capture sufficient transliteration information from X to produce correct strings in Y"*. We hypothesize that two parameters of the bridge language, namely, the orthography inventory and the phoneme-to-grapheme entropy, that has most influence on the quality of the transitional systems, and provide empirical evidence for this hypothesis.

5.1 Richer Orthographic Inventory

In each of the successful bridge systems (that is, those with a relative performance drop of less than 10%), presented in Table 1, namely, *Hin-Eng-Ara*, *Eng-Ara-Heb* and *Hin-Kan-Eng*, the bridge language has, in general, richer orthographic inventory than the target language. Arabic has a reduced set of vowels, and hence poorer orthographic inventory compared with English. Similarly, between the closely related Semitic languages Arabic-Hebrew, there is a many-to-one mapping from Arabic to Hebrew, and between Kannada-English, Kannada has nearly a superset of vowels and consonants as compared to English or Hindi.

As an example for a poor choice of Z, we present a transitional system, *Hindi → Arabic → English*, in Table 6, in which the transitional language z (Arabic) has smaller orthographic inventory than Y (English).

Arabic has a reduced set of vowels and, unlike English, in most contexts short vowels are optional. As a result, when Arabic is used as the bridge language the loss of information (in terms of vowels) is large

Language Pair	**ACC-1**	**Relative change in ACC-1**
Hin-Eng	0.422	
Hin-Ara-Eng	0.155	-64.28%

Table 6: Incorrect choice of bridge language

and the second stage system has no possibility of recovering from such a loss. The performance of the bridge system confirms such a drastic drop in ACC-1 of nearly 64% compared with the direct system.

5.2 Higher Phoneme-Grapheme Entropy

We also find that the entropy in phoneme - grapheme mapping of a language indicate a good correlation with a good choice for a transition language. In a good transitional system (say, *Hin-Eng-Rus*), English has a more ambiguous phoneme-to-grapheme mapping than Russian; for example, in English the phoneme ‘s’ as in *Sam* or *Cecilia* can be represented by the graphemes ‘c’ and ‘s’, whereas Russian uses only a single character to represent this phoneme. In such cases, the ambiguity introduced by the bridge language helps in recovering from errors in the $X \rightarrow Z$ system. The relative loss of ACC-1 for this transitional system is only about 8%. The Table 7 shows another transitional system, in which a poor choice was for the transitional language was made.

Language Pair	**ACC-1**	**Relative change in ACC-1**
Hin-Eng	0.422	
Hin-Tam-Eng	0.231	-45.26%

Table 7: Incorrect choice of bridge language

Tamil has a reduced set of consonants compared with Hindi or English. For example, the Hindi consonants (k, kh, g, gh) are represented by a single character in Tamil. As a result, when Tamil is used as the bridge language it looses information (in terms of consonants) and results in a significant drop in performance (nearly a 45% drop in ACC-1) for the bridge system.

6 Effectiveness of Bridge Transliteration on CLIR System

In this section, we demonstrate the effectiveness of our bridge transliteration system on a downstream application, namely, a Crosslingual Information Retrieval system. We used the standard document collections from CLEF 2006 (Nardi and Peters, 2006), CLEF 2007 (Nardi and Peters, 2007) and FIRE 2008 (FIRE, 2008). We used Hindi as the query language. All the three fields (title, description and narration) of the topics were used for the retrieval. Since the collection and topics are from the previous years, their relevance judgments were also available as a reference for automatic evaluation.

6.1 Experimental Setup

We used primarily the statistical dictionaries generated by training statistical word alignment models on an existing Hindi-English parallel corpora. As with any CLIR system that uses translation lexicon, we faced the problem of out-of-vocabulary (OOV) query terms that need to be transliterated, as they are typically proper names in the target language. First, for comparison, we used the above mentioned CLIR system with no transliteration engine (*Basic*), and measured the crosslingual retrieval performance. Clearly, the OOV terms would not be converted into target language, and hence contribute nothing to the retrieval performance. Second, we integrated a direct machine transliteration system between Hindi and English (*D-HiEn*), and calibrated the improvement in performance. Third, we integrate, instead of a direct system, a bridge transliteration system between Hindi and English, transitioning through Kannada (*B-HiKaEn*). For both, direct as well as bridge transliteration, we retained the top-5 transliterations generated by the appropriate system, for retrieval.

6.2 Results and Discussion

The results of the above experiments are given in Table 7. The current focus of these experiments is to answer the question of *whether the bridge machine transliteration systems used to transliterate the OOV words in Hindi queries to English* (by stepping through Kannada) *performs at par with a direct transliteration system*. As expected, enhancing the CLIR system with a machine transliteration sys-

Collection	CLIR System	MAP	Relative MAP change from *Basic*	Recall	Relative Recall change from *Basic*
CLEF 2006	Basic	0.1463	-	0.4952	-
	D-HiEn	0.1536	+4.98%	0.5151	+4.01%
	B-HiKaEn	0.1529	+4.51%	0.5302	+7.06%
CLEF 2007	Basic	0.2521	-	0.7156	-
	D-HiEn	0.2556	+1.38%	0.7170	+ 0.19%
	B-HiKaEn	0.2748	+9.00%	0.7174	+ 0.25%
FIRE 2008	Basic	0.4361	-	0.8457	-
	D-HiEn	0.4505	+3.30%	0.8506	+0.57%
	B-HiKaEn	0.4573	+4.86%	0.8621	+1.93%

Table 8: CLIR Experiments with bridge transliteration systems

tem (*D-HiEn*) gives better results over a CLIR system with no transliteration functionality (*Basic*). On the standard test collections, the bridge transliteration system performs in par or better than the direct transliteration system in terms of MAP as well as recall. Even though, the bridged system is of slightly lesser quality in ACC-1 in Hi-Ka-En, compared to Hi-En (see Table 1), the top-5 results had captured the correct transliteration, as shown in our analysis. A detailed analysis of the query translations produced by the above systems showed that in some cases the bridge systems does produce a better transliteration thereby leading to a better MAP. As an illustration, consider the OOV terms वेटिकन {`Vatican`} and नेस्ले {`Nestle`} and the corresponding transliterations generated by the different systems. The Direct-HiEn system was unable to

OOV term	D-HiEn	B-HiKaEn
वेटिकन (`vatican`)	vetican	vetican
	veticon	vettican
	vettican	**vatican**
	vetticon	watican
	wetican	wetican
नेस्ले (`nestle`)	nesle	**nestle**
	nesly	nesle
	nesley	nesley
	nessle	nestley
	nesey	nesly

Table 9: Sample output in direct and bridge systems

generate the correct transliteration in the top-5 results whereas the B-HiKaEn was able to produce the correct transliteration in the top-5 results thereby resulting in an improvement in MAP for these queries.

7 Conclusions

In this paper, we introduced the idea of bridge transliteration systems that were developed employing well-studied orthographic approaches between constituent languages. We empirically established the quality of such bridge transliteration systems and showed that quite contrary to our expectations, the quality of such systems does not degrade drastically as compared to the direct systems. Our error analysis showed that these better-than-expected results can be attributed to (i) Better performance ($\sim$10-12%) of the second stage system on the strings transliterated correctly by the first stage system and (ii) Significant ($\sim$25%) error correction in the second stage. Next, we highlighted that the performance of such bridge systems will be satisfactory as long as the orthographic inventory of the bridge language is either richer or more ambiguous as compared to the target language. We showed that our results are consistent with this hypothesis and provided two examples where there is a significant drop in the accuracy when the bridge language violates the above constraints. Finally, we showed that a state of the art CLIR system integrated with a bridge transliteration system performs in par with the same CLIR system integrated with a direct transliteration system, vindicating our claim that such bridge transliteration systems can be use in real-world applications to alleviate the resource requirement of ${}^{n}C_2$ parallel names corpora.

References

Peter E Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19:263–311.

FIRE. 2008. Forum for information retrieval evaluation.

Isao Goto, Naoto Kato, Noriyoshi Uratani, and Terumasa Ehara. 2003. Transliteration considering context information based on the maximum entropy method. In *Proceedings of MT-Summit IX*, pages 125–132.

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*, pages 383–389.

Byung-Ju Kang and Key-Sun Choi. 2000. Automatic transliteration and back-transliteration by decision tree learning. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, pages 1135–1411.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *Computational Linguistics*, pages 128–135.

John D. Lafferty, Andrew Mccallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA.

Jae Sung Lee and Key-Sun Choi. 1998. English to korean statistical transliteration for information retrieval. In *Computer Processing of Oriental Languages*, pages 17–37.

Haizhou Li, A Kumaran, , Min Zhang, and Vladimir Pervouvhine. 2009. Whitepaper of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 19–26, Suntec, Singapore, August. Association for Computational Linguistics.

Thomas Mandl and Christa Womser-Hacker. 2004. How do named entities contribute to retrieval effectiveness? In *CLEF*, pages 833–842.

Preslav Nakov and Hwee Tou Ng. 2009. Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1367, Singapore, August. Association for Computational Linguistics.

A Nardi and C Peters. 2006. Working notes for the clef 2006 workshop.

A Nardi and C Peters. 2007. Working notes for the clef 2007 workshop.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Jong-hoon Oh and Key-Sun Choi. 2002. An english-korean transliteration model using pronunciation and contextual rules. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 758–764.

Bonnie Glover Stalls and Kevin Knight. 1998. Translating names and technical terms in arabic text. In *Proceedings of COLING/ACL Workshop on Computational Approaches to Semitic Languages*, pages 34–41.

Raghavendra Udupa, K Saravanan, 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 *ECIR'09: Proceedings of the 31st European Conference on IR research on Advances in Information Retrieval*, pages 437–448, Toulouse, France.

Suryaganesh Veeravalli, Sreeharsha Yella, Prasad Pingali, and Vasudeva Varma. 2008. Statistical transliteration for cross language information retrieval using hmm alignment model and crf. In *Proceedings of the 2nd workshop on Cross Lingual Information Access (CLIA) Addressing the Information Need of Multilingual Societies*.

Hua Wu and Haifeng Wang. 2007. Pivot language approach for phrase-based statistical machine translation. *Machine Translation*, 21(3):165–181.