Character Sequence Modeling for Transliteration

Manoj Kumar Chinnakotla and **Om P. Damani**
Department of Computer Science and Engineering,
Indian Institute of Technology, Bombay
Mumbai, India
{manoj,damani}@cse.iitb.ac.in

Abstract

The Character Sequence Modeling (CSM), typically called the Language Modeling, has not received sufficient attention in the current transliteration research. We discuss the impact of various CSM factors like word granularity, smoothing technique, corpus variation, and word origin on the transliteration accuracy. We demonstrate the importance of CSM by showing that for transliterating into English, for two very different languages, Hindi and Persian, systems employing only monolingual resources and simple non-probabilistic character mappings achieve accuracy close to that of the baseline statistical systems employing parallel transliteration pairs. It shows that a reasonable transliteration system can be built for resource scarce languages that lack large parallel corpora.

1 Introduction

Transliteration is the process of mapping a written word from a language-script pair to another language-script pair. For example, Hindi word परम and शेयर corresponds to English words *param* and *share* respectively. Transliterating a word from the language of its origin to a foreign language is called *Forward Transliteration*, while transliterating a loan-word written in a foreign language back to the language of its origin is called *Backward Transliteration*. Our focus is on the general purpose transliteration into English from resource scarce languages for which very large parallel corpora of transliteration pairs does not exist. By general purpose, we mean that the system should do both forward and backward transliteration.

Given the wide variety of script-language pairs in the world, many different methods have been proposed for transliteration: grapheme based (Ganesh et al., 2008; AbdulJaleel and Larkey, 2003; Sherif and Kondrak, 2007; Haizhou et al., 2004; Kumaran and Kellner, 2007; Ekbal et al., 2006), phoneme based (Surana and Singh, 2008; Virga and Khudanpur, 2003; Knight and Graehl, 1997), and hybrid (Oh and Choi, 2002). Some of these systems are statistical in nature, while others are rule-based. One thing that is common to these systems is the direct or indirect use of the *Character Sequence Modeling (CSM)*, typically called the Language Modeling.

In transliteration literature (Surana and Singh, 2008; Kumaran and Kellner, 2007; Sherif and Kondrak, 2007), other than mentioning *N* for the N-gram model, many papers do not give other important details like the corpus over which the model was computed, back-off scheme employed, and what constitutes a *word* in such a model . While these factors are important for any application of language model, we argue that Character Sequence Modeling is different from traditional language modeling and its exploration can pay rich dividends.

We present the results for grapheme to grapheme based general purpose Hindi to English and Persian to English transliteration systems. Our systems use CSM on source side for word origin identification, a simple non-probabilistic character mapping to generate transliteration candidates, and then use CSM on the target side to rank the candidates. We demonstrate the importance of CSM by noting that our systems employing only monolingual resources and simple non-probabilistic character mapping achieve transliteration accuracy close to that of the baseline statistical systems based on parallel transliteration pairs. This shows that a reasonable transliteration system can be built for resource scarce languages that lack large parallel corpora required to train statistical systems. Besides, of particular interest, is the fact that none of the authors know Persian, a language known to present difficulties for transliteration since short vowels are typically not written down in the Persian words.

The organization of the paper is as follows: In Section 2, we formally define the Character Sequence Model (CSM). In Section 3, we present the details of our transliteration system like character mappings, candidate generation and pruning mechanism. Later, we also present the baseline approaches and the evaluation metrics used for comparison. In Section 4, we discuss the sequence of techniques used for improving the CSM performance and their impact on transliteration accuracy. In Section 5, we discuss the effect of word origin identification on improving CSM ranking performance. In Section 6, we present the results on the Hindi-English test set followed by error analysis. Section 7 presents the results on Persian-English dataset. Finally, Section 8 concludes the paper.

2 Character Sequence Modeling (CSM)

Phonotactic constraints dictate that a language should follow certain spelling patterns in the word formation. For example, in English, words like *hlad* and *mgla* are not valid because these sound formations are not natural in English. The aim of *CSM* is to learn these spelling patterns.

A *Character Sequence Model* for a language is a probability distribution over sequence of characters within a word. Let $W =< c_1, c_2, \ldots, c_n >$ be a word. Then with Order N Markov Assumption,

$$P(W) = \prod_{i=1}^{n} P(c_i \mid c_{i-1}, \ldots, c_{i-N})$$

The very first step in any language modeling work is the decision of what to model, i.e. which corpus to model. Most researchers have used the unique word list from the training data or some named entity list from the web as the corpus being modeled. We use English Wikipedia as our target corpus. The Wikipedia includes words and named entities from all parts of the world (5.6 million word forms in 2007, of which 1.8 million word forms had frequency greater than 2) and continues to grow exponentially. Hence it serves as a good general purpose corpus for transliterating into English.

3 Basic System

While our system architecture is common for both Persian to English and Hindi to English transliteration, we are not familiar with the Persian script. Therefore, we illustrate our system with the help of Hindi to English transliteration examples only. Towards the end of the paper, we present the results for Persian to English transliteration.

Hindi Character	**English Mapping**
क्	k,c,q,ck|!S,ch
ख्	kh,ck,k
ग्	g,gh
एक्स	x

Table 1: Hindi to English Constrained Rule Base Snippet (!S means not at the beginning)

Hindi words are written in Devanagari script while English words are written in Roman script. When there is no confusion, we use the terms *Devanagari word* and *Hindi word* interchangeably. Our system uses a character mapping table between the Hindi and English letters (snippet shown in the Table 1) and uses CSM for ranking the generated candidates. Most rules are of the form (ख् → *kh,ck,k*) where a single Hindi character ख् gets mapped to one or more English character sequences *kh, ck, k*. Some rules are of the form (एक्स→*x*) where a Hindi character sequence एक्स gets mapped to a single English character *x*. Combination of different mapping options for each character in the input Hindi word results in different transliteration candidates. For example consider the Hindi word दीपक (*deepak*). Note that each Devanagari consonant symbol that is not followed by a vowel represents that consonant plus an inherent *schwa vowel sound* अ (Shukla, 2000). Hence दीपक is processed as द्+ई+प्+अ+क्+अ. Since द्, ई, प्, अ, and क् have 2, 3, 1, 5 and 2 possible mappings respectively, hence a total of 2*3*1*5*2*5=300 transliteration candidates should be considered (Example: *deepak*, *dipaka*, *deepuk* etc.). Since the *schwa* vowel sound अ gets dropped in some contexts, we also add a NULL character mapping (no character appended during candidate generation) corresponding to *schwa* to

	Indo-Arabic Org.	Non Indo-Arabic Org.	Total	Purpose
Training	13666	10372	24038	Training CRF, SMT and Word Origin Identification
Development	2016	1365	3381	Tuning CSM parameters in Rule-Based Sys
Test	1422	1316	2738	Perf. Comparison of CRF, SMT, and Rule Based Sys
NEWS09TD Test	6366	4916	11282	Perf. Comparison of CRF, SMT, and Rule Based Sys

Figure 1: Hindi-English Dataset Details

	Overall Acc. (%)	Indo-Arabic Org. Acc. (%)	Non Indo-Arabic Org. Acc. (%)	MRR
Basic System (N=3)	27.9	25.6	31.3	0.164
Enlarge Alphabet (N=3)	32.9	29.6	37.8	0.203
Baseline Approaches				
CRF Transliteration	68.2	79.6	51.4	0.528
SMT Based Transliteration	75.5	84.7	61.8	0.593

Figure 2: Result Comparison of Basic System with Baseline Approaches

take care of this phenomenon. To avoid processing exponential number of candidates, we process the input characters one at a time and use CSM probability for pruning the partially generated candidates at each step. We train a trigram CSM on the unique words of Wikipedia using the SRILM toolkit (Stolcke, 2002) with the default options.

We randomly partition a parallel corpus of 30157 word pairs into training, development, and the test set, the details of which are given in the Figure 1. In addition to this, another test set called NEWS09TD (discussed in Section 6) is also used. Note that the word origin information is not used and discussed till Section 5, but we include it in Figure 1 for presenting all data related details in a single place.

3.1 Baseline Approaches for Comparison

The following language independent statistical approaches have recently been shown to perform quite well for the transliteration task. We chose them as baseline since they can be implemented easily using off-the-shelf components like CRF++ (Kudo, 2003), GIZA++ (Och and Ney, 2003) and Moses (Hoang et al., 2007). Both systems are trained with a parallel corpus of 24,000 transliteration pairs.

3.1.1 CRF Based Transliteration

(Ganesh et al., 2008) proposed a Conditional Random Field (CRF) based approach. They pose the problem as a sequence learning problem given the input Devanagari string and report better accuracies than the HMM based approach (AbdulJaleel and Larkey, 2003). We implemented a modified version of their approach using CRF++.

3.1.2 SMT Based Transliteration

In (Sherif and Kondrak, 2007) and (Huang, 2005) the transliteration problem is posed as a Phrase Based Machine Translation problem where the words are replaced by the characters. We implemented the above approach using GIZA++ aligner, the Moses decoder, and the SRILM toolkit. As discussed in the next section, we use N=5 and the enlarged alphabet for the language modeling. SRILM and Moses were used with the default options, except that the *distinct* option was also given to avoid any duplicates in the output list.

3.2 Evaluation Metrics

A transliteration system produces a ranked list of possible answers. Given a parallel corpus containing a list of M source language words and a set of possible target language transliterations for each source word, we define the answer for the given source word to be correct at rank *k*, if one of the top-*k* candidates generated by the system belongs to the set of corresponding transliterations. We also consider *Mean Reciprocal Rank (MRR)* which is defined as follows:

$$MRR = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{Rank(i)}$$

where $Rank(i)$ is the rank of the first correct answer for the i^{th} input word. We report the Accuracy and MRR figures at rank 5 throughout the paper, unless otherwise stated.

4 Improving the CSM Performance

Figure 2 shows the performance of our basic system and that of the baseline systems. Both the baseline systems using parallel corpus outperform our basic system with a huge margin.

4.1 Enlarging the Alphabet

In our system, since one Hindi character can map to multiple English characters, the candidates generated are of varying length, and the CSM assigns much lower probability to the longer candidates than to the shorter ones. For example, for word फाटक (*phatak*), candidate *fatak* is assigned more weight compared to *phatak* because letter p and h are treated as two different graphemes, even though both *f* and *ph* correspond to the same Hindi letter फ. We solve this problem by changing the gram (basic unit) in our N-Gram model. We treat several multi-character sequences as a single character. In (Ekbal et al., 2006), the alphabet is extended by extracting multi-character sequences derived from an alignment of parallel corpus. Since we are not using parallel corpora, we add multi-characters gram based on:

- Common Digraphs: As in (Teahan, 1997), we treat frequently occurring digraphs like *sh*, *ck* etc. as a single character.
- Double letters: Typically, a pair of identical letters is pronounced as a single phoneme in English, and hence we add them to our alphabet. For example, *ll* and *ss* in *mill* and *miss*.
- Schwa handling: In Section 3, we discussed that in written words, each Devanagari consonant that is not followed by a vowel contains an inherent schwa vowel which may or may not be pronounced. For a simple rule base system, it is hard to decide when a schwa will be pronounced and when will it be deleted. Hence we treat basic consonants followed by schwa to be a single letter. Example, *ka, kha, ga* etc.

As shown in Figure 2, this enlarging of alphabet improves the performance of system from 27.9% to 32.9%. As an example, candidates for word इम्प्रेस (*impress*) changes from {*impres*, *empres*, *empras*, *imprec*} in the basic system to {*impres*, *empres*, ***impress***, *empress*, *impras*}, thus including the correct answer with *ss*.

4.2 Varying N

So far we have been using the trigram model because that is the default in the SRILM toolkit, and it is the model used by many transliteration researchers. We next vary the value of '*N* and find that the N=5 gives us the best results and this increases the accuracy to 47.7%. For example, candidates for word डिज़ॉल्व (*dissolve*) changes from {*dicallo*, *dicalo*, *decolo*, *disolo*, *dicalva*} to {*dissolo*, ***dissolve***, *decolo*, *disallo*, *desalu*} taking benefit of *d i ss o l* as a 5-gram unit in the training corpus.

4.3 Weighing Each Word

Till now, we have been using the list of unique words from the Wikipedia as our corpus. A list of unique words gives equal weightage to a correct spelling and a misspelling, or a rare spelling variation. Hence, we bias the CSM in favor of the more prevalent spelling by using word-frequency as the weight. Given that many words occur with very high frequency, such a model heavily favors the common words. Hence we use the natural logarithm of the word frequency (rounded down), as the weight. As a desirable side-effect, words occurring only once or twice are not used at all and the number of unique word forms in the corpus reduces from 5.6 million to 1.8 million. This corpus weighing increases the system accuracy to 53.6%. A English word like डैनिश (*danish*) benefits from it with the candidate set changing from {*dennis*, *danes*, *denis*, *denes*, *danese*} to {*denis*, ***danish***, *danes*, *dennis*, *danese*}.

4.4 Smoothing and Discounting Techniques

The experiments so far have used the default SRILM smoothing, which is the Katz Backoff with Good-Turing Discounting (Jurafsky and Martin, 2008). While this is known to work well for word sequence modeling tasks, CSM requires a different smoothing method. Number of characters in a given alphabet is orders of magnitude smaller than the number of words in a language. Hence the number of N-Grams to be considered in a transliteration application is far less than that in a word modeling task. Also, the number of distinct sentences a system may have to process is infinite, while the number of possible words is arguably finite. Hence the smoothing technique for the CSM should give less weightage to the unseen N-Grams than what is typically done in the word modeling tasks.

We compare the performance of Katz Backoff and Good-Turing Discounting with two other methods: Chen-Goodman Backoff with Kneser-Ney Discounting (implemented in the SRILM toolkit) (Jurafsky and Martin, 2008), and PPM-D smoothing (not implemented in SRILM) (Teahan, 1997). Since methods implemented in SRILM toolkit are widely known in NLP community, we do not describe them here. We only explain the

	Overall Acc. (%)	Indo-Arabic Org. Acc. (%)	Non Indo-Arabic Org. Acc. (%)	MRR	Invocab Acc. (%) (Total Invocab: 2564)	OOV Acc. (%) (Total OOVs: 836)
GT Disc. + Katz BO (N=5)	53.6	42.1	70.5	0.391	66.5	14.0
CG Disc. + Kneser Ney (N=5)	44.8	38.8	53.7	0.293	54.1	16.3
PPM-D (N=6)	**58.7**	44.0	80.1	**0.457**	75.2	7.9
MLE (N=6)	43.9	24.8	71.9	0.345	57.8	1.0
Common Rules + Diff. CSM + PPM-D	67.8	60.6	78.3	0.544	80.4	29.0
Diff. Rules + Diff. CSM + PPM-D	**74.1**	71.2	78.4	**0.615**	80.8	53.8

Figure 3: Hindi-English Development Set Results: Effect of Smoothing and Word Origin Identification

PPM-D smoothing which is relatively unknown in the NLP community.

Prefix Based Partial Match (PPM) is an adaptive N-gram character sequence model well-known in the text compression community (Cleary et. al., 1984). Let a be the context observed so far and z be the next symbol in sequence. Let $c(z)$ be the number of times the context a was followed by the symbol z and n ($\sum_k c(k)$) be the total number of symbols that have followed a; Let t be the number of distinct symbols that have followed context a so far and M be the total number of distinct symbols seen in the training data. We use the PPM-D smoothing method (Teahan, 1997) which is given by:

$$Pr(z \mid a) = \begin{cases} \frac{2 \cdot c(z) - 1}{2 \cdot n} & \text{If } c(z) > 0 \text{ in context } a \\ \frac{1}{M-t} \cdot \frac{t}{2 \cdot n} & \text{Otherwise} \end{cases}$$

Since we want to give much less weightage to the unseen N-Grams, a natural baseline method is to use Maximum Likelihood Estimate (MLE), and employ no smoothing. We also present the comparison results for this option.

4.5 System Performance

The results of applying various smoothing techniques are shown in the Figure 3. To put the results in perspective, we have categorized them by whether the correct English transliteration is present in the corpus being modeled or not. The Kneser-Ney technique gives the best result for the out-of-vocabulary (words which are not in Wikipedia) target words while the PPM-D smoothing gives the best result for the in-vocabulary words. The performance of the PPM-D is not unexpected since it concentrates the probability mass heavily on the seen events. The choice of the smoothing technique depends on whether we expect to see lot of out-of-vocabulary words in our application or not. Hence for a general purpose system with Wikipedia as the target corpus, we expect to run into the in-vocabulary words much more often than the out-of-vocabulary words. Therefore, we chose PPM-D smoothing. With this choice, a seen word like कुशीनगर (kushinagar) breaks into the top-5 with the candidate set changing from $\{$*choosinger*, *cusinger*, *cussinger*, *cousinger*, *kusinger*$\}$ to $\{$***kushinagar***, *kusinagara*, *cousinger*, *kuchinger*, *cusinger*$\}$ due to the less weightage given to unseen sequences like *choosinger*, *cusinger* etc.

5 Word Origin Identification

It is observed in (Huang, 2005; Surana and Singh, 2008) that identifying the word origin is critical to the transliteration success. In (Huang, 2005), a statistical clustering technique is employed on a parallel transliteration corpus, and 50 different classes of English words are created. Since our work focuses on the use of monolingual resources, ideally, we should classify words based on the phonological typology. In the absence of resources to do such a classification, we simply classify words by whether they are of Indian origin or not. In (Shukla, 2000) the origin of Hindi words is traced to four sources: a) Native words (Sanskrit words and their derivatives), b) Persian, Arabic, and Turkish words, c) English words, and d) Portuguese words. Since not only the words of other Indian languages, but even the words of Persian (Ex: जमीन *zameen*), Arabic (Ex: औरत *aurat*), and Turkish (Ex: दरोगा *daroga*) sound like Hindi words to native speakers, we finally classified words by whether they are of Indo-Arabic origin or not. Note that in the context of resource scarce languages, effort required to manually annotate the origin of a given word is much less than

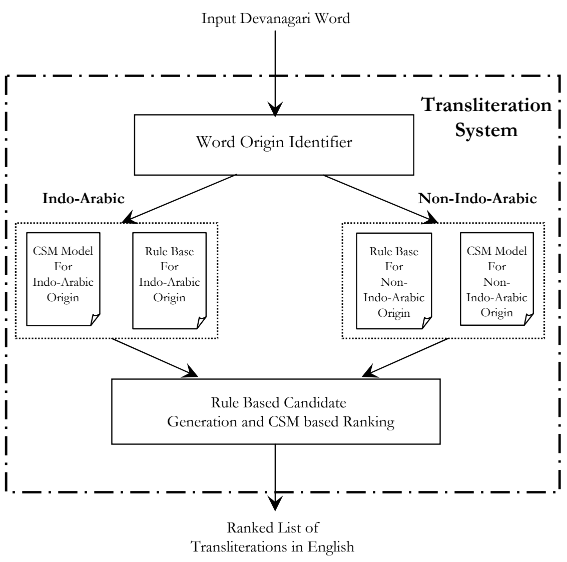
Input Devanagari Word
Transliteration System
Word Origin Identifier
Indo-Arabic
Non-Indo-Arabic
CSM Model For Indo-Arabic Origin
Rule Base For Indo-Arabic Origin
Rule Base For Non-Indo-Arabic Origin
CSM Model For Non-Indo-Arabic Origin
Rule Based Candidate Generation and CSM based Ranking
Ranked List of Transliterations in English

Figure 4: System Architecture with Word Origin Identification Module

that of producing the correct transliteration.

Previously, N-Gram models have been used for the word origin identification in (Llitjos and Black, 2001) and (Surana and Singh, 2008). We use a similar model where we learn two different CSM classes corresponding to each origin and assign a word to the class which gives it higher probability. The training data consists of 24,000 Hindi words manually annotated with the origin. In order to determine the ideal N-Gram size, the same 24000 words were used as the development set as well. The N-Gram size 3 gives the best accuracy of 87.6% for our chosen training and development set. In (Surana and Singh, 2008), 5-grams are found to be most effective for identifying the origin of words written in the Roman script. In contrast, we find that 3-grams are best suited for the origin identification of written Hindi words. This is because many consonants in written Devanagari words include an inherent schwa vowel, as discussed in Section 3.

5.1 Origin Dependent CSM

The purpose of word origin identification is to use a different, origin dependent CSM on the target side. For training the English CSM for the words of Indo-Arabic origin, a collection of 167,814 Indo-Arabic names written in the Roman script were used. For Non-Indo-Arabic origin words, the English Wikipedia corpus was used. While Wikipedia also contains many words of Indo-Arabic origin, words of Non-Indo-Arabic origin are orders of magnitude more frequent.

The system works as before, except that the character sequence model for ranking the candidates is now based on the origin assigned by the classifier as depicted in Figure 4. As shown in Figure 3, the accuracy of the system now jumps to 68.5%, slightly exceeding that of the CRF. Note the particular increase in the accuracy of Indo-Arabic origin words, since the language model for them gets quite altered. As an example, बोधगया (*bodhgaya*) benefits from the changed CSM where the prefix *b o dh* gets preference over the prefix *b o d* and the candidates change from {*bodgaya*, *bodgia*, *bodegua*, *bodeguo*, *bodegea*} to {*bodhagaya*, *bodhagya*, *bodhgya*, ***bodhgaya***, *bodhegya*}. For non-Indo-Arabic origin words the CSM has not really changed much. Compared to results shown in Figure 3, the reduction in the accuracy of foreign origin words is due to misclassification errors - it is fatal for a non Indo-Arabic word to be classified as that of Indo-Arabic origin. For example, when सन्निवेल (*sunnyvale*) gets misclassified as Indo-Arabic origin, its candidates change from {***sunnyvale***, *snivel*, *snivell*, *snivelle*, *conewall*} to {*sanewal*, *shanewal*, *sanewala*, *shanewala*, *shaneevala*}.

5.2 Origin Dependent Rule Base

The error analysis at this stage still shows scope for improvement. Regardless of the word origin, same set of candidates are currently being generated. Hence we decided to use different character mapping depending on the word origin. This step is arguably tricky. We cannot characterize our system as a *simple one* anymore. In fact, the mapping rules were improved using trial-and-error. Despite having these misgivings, we think that such a step still helps us show the power of CSM. The system guesses the origin of the input Hindi word, generates candidates from the appropriate mapping table based on the word origin, and ranks the generated English candidates using the appropriate CSM. The performance of this system is shown in Figure 3. Its performance is comparable to that of the SMT based system. The major beneficiaries of Origin Dependent Rule Base are words of Indo-Arabic origin. For example, क़ maps to only *k* and not to *ch* in the Indo-Arabic origin rule base and hence the candidates for कामराव (*kamrao*) changes from {*chamaroo*, *kamaroo*, *kamar-*

	Overall Acc. (%)	Indo-Arabic Org. Acc. (%)	Non Indo-Arabic Org. Acc. (%)	MRR
Test (2740 Words)				
CRF Transliteration	64.7	79.0	49.2	0.490
SMT Transliteration	71.4	81.2	60.9	0.546
PPM-D (N=6)	**78.0**	79.7	76.0	**0.665**
NEWS09TD Dataset				
CRF Transliteration	62.2	72.0	49.5	0.467
SMT Transliteration	61.7	68.9	52.4	0.460
PPM-D (N=6)	**67.8**	60.6	78.3	**0.578**

Figure 5: Hindi to English Test Set Results

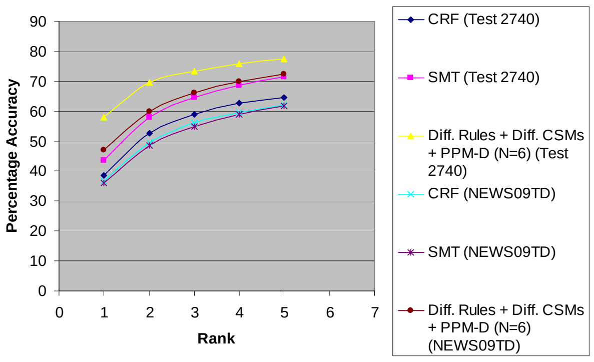

Figure 6: Hindi to English Test Set Rank vs. Accuracy

ava, *chamarava*, *komrao*} to {*kamarava*, ***kamrao***, *kamarav*, *kaamrao*, *kamerava*}.

6 Test Set Results

Till now we have been optimizing our system based on its performance on the development data. We next experiment with two different test sets. Our first test set is the held-out set of 3400 words. This dataset was selected at the time when there was no publicly available test set. Later a parallel list of 9975 pairs for training and 1000 pairs for development were released as part of the NEWS 2009 workshop Shared Task on English-Hindi Transliteration[1] (Li et al., 2009). Since the test set size of NEWS 2009 was only 1000, we decided to use the training and development set of NEWS 2009 workshop as our second test set and we refer it as NEWS09TD (NEWS 2009 Training and Development Sets) test set. Also, the NEWS data is for English to Hindi transliteration, with multiple Hindi transliterations for some English word. When we reverse the parallel corpus direction, treating each of the multiple Hindi targets as separate test item increases the test set size to 11283.

The results for the two test sets are shown in Figure 5 and Figure 6. The performance of the CRF and our system does not vary a lot between the development set and the two test sets. In contrast, for the SMT system, results on the held-out test set are comparable to the results on the development set, but the results on the NEWS09TD test set are worse. As observed in (Karimi et al., 2007), different set of human annotators use slightly different conventions when transliterating, and hence the SMT based system does much worse when the test data does not come from the same source as the training and the development data.

[1]https://translit.i2r.a-star.edu.sg/news2009/

6.1 A Comment on the Reported Accuracies

Currently researchers report widely varying number for transliteration accuracy - from 32%-88% (Karimi et al., 2007). For Hindi-to-English transliteration, (Ganesh et al., 2008) report 72.1% accuracy at rank 5 for a CRF based system, while (Kumaran and Kellner, 2007) have reported 31.1% accuracy at rank 10.

Not too much should not be read into these comparisons. We have reported several accuracy figures - that for words of Indo-Arabic origin, words of non Indo-Arabic origin, in-vocabulary words, and out-of-vocabulary words. By suitably changing the mix of such categories in the test set, the accuracy numbers can be correspondingly modified. Hence comparing results on dissimilar datasets is not very meaningful. Ideally researchers should report the numbers for each of these categories.

Another factor affecting the results is the quality of the test set, as discussed in (Karimi et al., 2007). On a different parallel name list of 5000 Indian names[2], our system gives accuracy of 92% at rank 5. But this list seems to have been generated by using a simple mapping table and does not represent the real life spelling variations and idiosyncrasies, and does not handle the schwa deletion phenomenon. Hence results reported on non-open or non-standard datasets should be taken with a grain of salt. Therefore, we make all our training, testing, and development data publicly available[3].

[2]http://www.cse.iitb.ac.in/~manoj/Indian-Names-Dataset.tar.gz

[3]http://www.cse.iitb.ac.in/~manoj/Hindi-Persian-Dataset.tar.gz

6.2 Error Analysis

There are several reasons for the errors in the output:

- **Multiple Transliterations:** For many words of non-Indian origin, multiple English words correspond to a given Hindi word and our system fails to guess the one in the Gold Standard. For example, for गेट्स (*gets, gates*), top 5 results from our system are: {*gets*, *ghats*, *getz*, *geths*, *geats*}, whereas the gold standard only contains *gates*.

- **Origin Misclassification:** As discussed in Section 5.1, origin misclassification is one of the important sources of error.

- **Lack of Context Sensitive Mapping Rules:** While the previous two causes of errors are system errors, this one is a model error. Only context we use is whether a character is at the beginning of a word or at the end of a word, and whether or not it follows a vowel. As a result our top five candidates for शेयर (*share*) are *chair*, *cheer*, *seer*, *sheer*, and *sauer*, whereas a systems with rules like श followed by a य gets transliterated as *sh* only might give the correct answer.

- **Schwa Related Errors:** Our system cannot distinguish between two very closely related words like धडकन (*dhadkan*) and धडक (*dhadak*) due to schwa deletion phenomena discussed earlier. Although the context of letter ड (*da*) is same in both cases, the schwa is dropped in one case and not in another.

  Note that since Transliteration is one of those problems where the accuracy of an answer is not defined solely by algorithmic rules but also by human convention, which can be inconsistent and somewhat ad-hoc at times. Therefore, any rule-based system is bound to have certain limitations.

7 Persian to English Transliteration

To further demonstrate the validity of our approach, we decided to experiment with some non-Indian language. Our only criterion for selecting the language was that there should be a publicly available character mapping into English, and a parallel transliteration corpus. We found that such resources are available for Persian to English

	Perso-Arabic Org.	Non Perso-Arabic Org.	Total (19940)
Training	2295	11645	13940
Development	509	2491	3000
Test	483	2517	3000

Figure 7: Persian Dataset Details

	Overall Acc. (%)	Perso-Arabic Org. Acc. (%)	Non Perso-Arabic Org. Acc. (%)	MRR
Basic System (N=3)	24.8	24.6	24.9	0.134
Enlarging Alphabet (N=3)	25.3	23.8	25.7	0.139
Word Weighing (N=3)	25.2	21.6	25.9	0.138
Baseline Approaches				
CRF	48.3	50.9	47.8	0.318
SMT	67.0	81.6	63.9	0.502

Figure 8: Persian-English Basic System and Baseline Approaches

transliteration in (Karimi, 2008). Hence, we decided to work on it, despite the fact that none of the authors have any knowledge of the Persian whatsoever (we cannot even read the Persian script). We also discovered that short vowels are typically absent from the written Persian words (Karimi, 2008), thus making the problem even more challenging. We obtained the Persian-English parallel corpus of 19940 word pairs from (Karimi, 2008). We randomly partitioned the data into training, development, and test set, details of which are given in Figure 7.

Persian Character	English Mapping
ا	a,aa,au
ف	f,ph
ی	y,ei,ay,ai,i,ee
س	c,s
ش	sh

Table 2: Persian to English Rule Base Snippet

7.1 Persian to English Character Mapping

We augmented the rule base obtained from (Karimi, 2008) slightly with the help of English to IPA (International Phonetic Alphabet) and Persian to IPA mappings. If an English character sequence and a Persian character sequence map to same IPA

	Overall Acc. (%)	Perso-Arabic Org. Acc. (%)	Non Perso-Arabic Org. Acc. (%)	MRR	Invocab Acc. (%) (Total Invocab: 2484)	OOV Acc. (%) (Total OOVs: 418)
GT Disc + Katz BO (N=6)	35.1	28.5	36.5	0.227	39.4	9.6
CG Disc + Kneser-Ney (N=7)	29.5	20.2	31.4	0.192	33.2	7.2
PPM-D (N=7)	**49.7**	45.4	50.7	**0.359**	56.8	7.2
MLE (N=7)	36.3	30.0	37.7	0.264	41.9	3.1

Figure 9: Persian to English Development Set Results

	Overall Acc. (%)	Perso-Arabic Org. Acc. (%)	Non Perso-Arabic Org. Acc. (%)	MRR
CRF Transliteration	48.0	48.8	47.8	0.325
SMT Based Transliteration	**67.2**	80.2	64.7	**0.502**
Rule Based Sys + PPM-D CSM	47.0	41.8	48.1	0.343

Figure 10: Persian to English Test Set Results

symbol, we mapped these sequences to each-other, for example ی and *ei, ay, ai, ee*. A segment of the mapping table is shown in the Table 2. Also, since short vowels are dropped from written Persian words, if a Persian consonant is not followed by a vowel, then we insert a short vowel after the consonant. For example, consider the word فرح (*farrah*). The first character ف (*f*) (note that Persian is read from right to left) is followed by another consonant ر (*r*). Since ف gets mapped to *f*, *ph*, we also generate *fa, fe, fi, fo, fu, pha, phe, phi, phu, pho* as candidates for ف. This results in a large number of spurious candidates. Many of the spurious insertions are ruled out by CSM and hence despite this added disadvantage, our system provides performance comparable to statistical systems, as discussed next.

7.2 Experiments and Results

We repeated the step-by-step procedure described earlier for Hindi to English transliteration. Since the Persian to English character mapping table is not separated by word origin, we could not do the word origin based experiments. The results of our experiments are shown in Figures 8, 9 and 10. The results follow the trend for Hindi to English transliteration as discussed in Section 4. The performance of our system is comparable to the CRF system while the SMT system does much better. It distinctly leaves open the possibility that the use of an origin dependent CSM on an independent test data will provide performance competitive with that of the SMT system.

Our results are of course much poorer than the state of the art results for Persian. For words of Perso-Arabic origin, a top-5 accuracy of 72.7% is reported for the best performing single model CV-MODEL1 on BP2E dataset (Karimi, 2008). But many of the factors discussed in this paper have not been attended to in (Karimi, 2008) and paying proper attention to CSM might help furthering of the state of the art for Persian to English transliteration.

8 Conclusions

In keeping with the general trend in NLP, almost all of the recent research in transliteration makes use of parallel corpora. In this paper, we show that by properly harnessing the monolingual resources, one can achieve transliteration accuracy that comes close to that of baseline statistical systems based on parallel corpora. This demonstrates that a reasonable transliteration system can be built for resource scarce languages which lack large parallel corpora by employing monolingual resources. Of course, the use of the CSM is not an end but just a beginning for these languages.

Acknowledgements

The first author was supported by a fellowship award from Infosys Technologies Limited. We would to thank Dr. Sarvnaz Karimi for providing us the Persian-English dataset and also the rule base and Dr. Maryam Shojaei Baghini for helping us in understanding some issues related to Persian script. Thanks are due to the Cross Lingual Information Access (CLIA) consortia project funded by the Department of Information Technology, Government of India and especially the Search and Information Extraction Lab (SIEL), IIIT Hyderabad for providing us a part of the parallel Hindi-

English resources used for evaluation. Finally, we would like to thank anonymous reviewers of an earlier version of this paper for their valuable comments which has helped in improving the content and presentation of this work.

References

[AbdulJaleel and Larkey, 2003] Nasreen AbdulJaleel and Leah S. Larkey. 2003. Statistical Transliteration for English-Arabic Cross Language Information Retrieval. In *CIKM '03:*, pages 139–146.

[Cleary et. al., 1984] John G. Cleary, Ian, and H. Witten. 1984. Data Compression using Adaptive Coding and Partial String Matching. *IEEE Transactions on Communications*, 32:396–402.

[Ekbal et al., 2006] Asif Ekbal, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. 2006. A Modified Joint Source-Channel Model for Transliteration. In *Proceedings of the COLING/ACL*, pages 191–198.

[Ganesh et al., 2008] Surya Ganesh, Sree Harsha, Prasad Pingali, and Vasudeva Verma. 2008. Statistical Transliteration for Cross Language Information Retrieval using HMM Alignment and CRF. In *Proceedings of Workshop on CLIA, Addressing the Needs of Multilingual Societies, IJCNLP*.

[Haizhou et al., 2004] Li Haizhou, Zhang Min, and Su Jian. 2004. A Joint Source-Channel Model for Machine Transliteration. In *ACL '04:*, page 159.

[Hoang et al., 2007] Hieu Hoang, Alexandra Birch, Chris Callison-burch, Richard Zens, Rwth Aachen, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondej Bojar. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of ACL, Demonstration Session*, pages 177–180.

[Huang, 2005] Fei Huang. 2005. Cluster-Specific Named Entity Transliteration. In *HLT '05:*, pages 435–442.

[Jurafsky and Martin, 2008] D. Jurafsky and J. H. Martin. 2008. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition, Second ed. Prentice Hall.

[Karimi et al., 2007] Sarvnaz Karimi, Andrew Turpin, and Falk Scholer. 2007. Corpus Effects on the Evaluation of Automated Transliteration Systems. In *Proceedings of ACL 2007*, pages 640–647.

[Karimi, 2008] Sarvnaz Karimi. 2008. *Machine Transliteration of Proper Names between English and Persian*. Ph.D. thesis, RMIT University, Australia.

[Knight and Graehl, 1997] Kevin Knight and Jonathan Graehl. 1997. Machine Transliteration. In *Proceedings of ACL 1997*, pages 128–135.

[Kudo, 2003] Kudo. 2003. CRF++: Yet Another CRF Toolkit.

[Kumaran and Kellner, 2007] A. Kumaran and Tobias Kellner. 2007. A Generic Framework for Machine Transliteration. In *SIGIR '07:*, pages 721–722.

[Li et al., 2009] H. Li, A. Kumaran, V. Pervouchine, and M. Zhang. 2009. Report on NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009).*

[Li et al., 2009] H. Li, A. Kumaran, M. Zhang and V. Pervouchine. 2009. Whitepaper of NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009).*

[Llitjos and Black, 2001] Ariadna Font Llitjos and Alan W. Black. 2001. Knowledge of Language Origin Improves Pronunciation Accuracy of Proper Names. In *Proceedings of EUROSPEECH '01*, pages 1919–1922.

[Och and Ney, 2003] Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

[Oh and Choi, 2002] J. H. Oh and K.S. Choi. 2002. An English-Korean Transliteration Model using Pronounciation and Context Rules. In *Proceedings of ACL*, pages 1–7.

[Sherif and Kondrak, 2007] Tarek Sherif and Grzegorz Kondrak. 2007. Substring-Based Transliteration. In *In Proceedings of ACL 2007*. The Association for Computer Linguistics.

[Shukla, 2000] Shaligram Shukla. 2000. *Hindi Phonology*. Lincom Studies in India-European Linguistics, Lincom Europa.

[Stolcke, 2002] Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *In Proceedings of Intl. Conf. on Spoken Language Processing*.

[Surana and Singh, 2008] Harshit Surana and A. K. Singh. 2008. A More Discerning and Adaptable Multilingual Transliteration Mechanism for Indian Languages. In *Proceedings of IJCNLP '08*.

[Teahan, 1997] W. J. Teahan. 1997. *Modelling English Text*. Ph.D. thesis, The University of Waikato, New Zealand.

[Virga and Khudanpur, 2003] Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of Proper Names in Cross-Lingual Information Retrieval. In *Workshop On Multilingual And Mixed-Language Named Entity Recognition*.