Transliteration Systems Across Indian Languages Using Parallel Corpora

Rishabh Srivastava and **Riyaz Ahmad Bhat**
Language Technologies Research Center
IIIT-Hyderabad, India
{rishabh.srivastava, riyaz.bhat}@research.iiit.ac.in

Abstract

Hindi is the lingua-franca of India. Although all non-native speakers can communicate well in Hindi, there are only a few who can read and write in it. In this work, we aim to bridge this gap by building transliteration systems that could transliterate Hindi into at-least 7 other Indian languages. The transliteration systems are developed as a reading aid for non-Hindi readers. The systems are trained on the transliteration pairs extracted automatically from a parallel corpora. All the transliteration systems perform satisfactorily for a non-Hindi reader to understand a Hindi text.

1 Introduction

India is home to languages from four language families namely Indo-Aryan, Dravidian, Austroasiatic and Tibeto-Burman. There are 22 official languages and more than 1000 dialects, which are written in more than 14 different scripts[1] in this country. Hindi, an Indo-Aryan language, written in Devanagari, is the lingua-franca of India (Masica, 1993, p. 6). Most Indians are orally proficient in Hindi while they lack a good proficiency in reading and writing it. In this work, we come up with transliteration systems, so that non-native speakers of Hindi don't face a problem in reading Hindi script. We considered 7 Indian languages, including 4 Indo-Aryan (Punjabi, Gujarati, Urdu and Bengali) and 3 Dravidian (Telugu, Tamil and Malayalam) languages, for this task. The quantity of Hindi literature (especially online) is more than twice as in any other Indian language. There are approximately 107 newspapers[2], 15 online newspapers[3] and 94067 Wikipedia articles[4] (reported in March 2013), which are published in Hindi. The transliteration systems will be helpful for non-Hindi readers to understand these as well as various other existing Hindi resources.

As the transliteration task has to be done for 7 languages, a rule-based system would become very expensive. The cost associated with crafting exhaustive rule-sets for transliteration has already been demostrated in works on Hindi-Punjabi (Goyal and Lehal, 2009), Hindi-Gujarati (Patel and Pareek, 2009) and Hindi-Urdu (Malik et al., 2009; Lehal and Saini, 2010). In this work, we have modelled the task of transliteration as a noisy channel model with minimum error rate training (Och, 2003). However, such a statistical modelling needs an ample amount of data for training and testing. The data is extracted from an Indian language sentence aligned parallel corpora available for 10 Indian languages. These sentences are automatically word aligned across the languages. Since these languages are written in different scripts, we have used an Indian modification of the soundex algorithm (Russell and Odell, 1918) (henceforth Indic-Soundex) for a normalized language representation. Extraction of the transliteration pairs (two words having the similar pronunciation) is then followed by Longest Common Subsequence (henceforth LCS) algorithm, a string similarity algorithm. The extracted pairs are evaluated manually by annotators and the accuracies are calculated. We found promising results as far as the accuracies of these extracted pairs are concerned. These transliteration pairs are then used to train the transliteration systems. Various evaluation tests are performed on these transliteration systems which confirm the high accuracy of these transliteration systems. Though the best system was nearly 70% accurate on word-level, the character-level accuracies (greater than 70% for all systems) along with the encouraging results from the human evaluations, clearly show

[1] http://en.wikipedia.org/wiki/Languages of India
[2] http://en.wikipedia.org/wiki/List_of_newspapers_in_India
[3] http://www.indiapress.org/
[4] http://stats.wikimedia.org/EN_India/Sitemap.htm

that these transliterations are good enough for a typical Indian reader to easily interpret the text.

1.1 Related Work

Knight (1998) provides a deep insight on how transliteration can be thought of as translation. Zhang et al.(2010) have proposed 2 approaches, for machine transliteration among English, Chinese, Japanese and Korean language pairs when extraction/creation of parallel data is expensive. Tiedemann (1998) has worked on text-based multi-language transliteration exploiting short aligned units and structural & orthographic similarities in a corpus. Indirect generation of Chinese text from English transliterated counter-part (Kuo and Yang, 2004) discusses the changes that happen in a borrowed word. Matthews (2007) has created statistical model for transliteration of proper names in English-Chinese and English-Arabic.

As Indian languages are written in different scripts, they must be converted to some common representation before comparison can be made between them. Grapheme to Phoneme conversion (Pagel et al., 1998) is one of the ways to do this. Gupta et al. (2010) have used WX notation as the common representation to transliterate among various Indian languages including Hindi, Bengali, Punjabi, Telugu, Malayalam and Kannada. Soundex algorithm (Russell and Odell, 1918) converts words into a common representation for comparison. Levenshtein distance (Levenshtein, 1966) between two strings has long been established as a distance function. It calculates the minimum number of insertions, deletions and substitutions needed to convert a string into another. Longest Common Subsequence (LCS) algorithm is similar to Levenshtein distance with the difference being that it does not consider substitution as a distance metric.

Zahid et al. (2010) have applied Soundex algorithm for extraction of English-Urdu transliteration pairs. An attempt towards a rule based phonetic matching algorithm for Hindi and Marathi using Soundex algorithm (Chaware and Rao, 2011) has given quite promising results. Soundex has already been used in many Indian language systems including Named entity recognition (Nayan et al., 2008) and cross-language information retrieval (Jagarlamudi and Kumaran, 2008). Although they applied soundex after transliteration from Indian language to English. Named-entity transliteration pairs mining from Tamil and English corpora has been performed earlier using a linear classifier (Saravanan and Kumaran, 2008). Sajjad et al. (2012) have mined transliteration pairs independent of the language pair using both supervised and unsupervised models. Transliteration pairs have also been mined from online Hindi song lyrics noting the word-by-word transliteration of Hindi songs which maintain the word order (Gupta et al., 2012).

In what follows, we present our methodology to extract transliteration pairs in section 2. The next section, Section 3, talks about the details of the creation and evaluation of transliteration systems. We conclude the paper in section 4.

2 Extraction of transliteration pairs

We first align the words for all the languages with Hindi in the parallel corpora. Phoneme matching techniques are applied to these pairs and the pairs satisfying the set threshold are selected. Given these pairs, transliteration systems are trained for all the 7 language pairs with Hindi as the source language.

2.1 Corpora

We have used the ILCI corpora (Jha, 2010) which contains 30000 parallel sentences per language for 11 languages (we have not considered English. Neither are Marathi and Konkani as the latter 2 are written in Devanagari script, which is same for Hindi). The corpora contains sentences from the domain of tourism and health with Hindi as their source language. Table 1 shows the various scripts in which these languages are written. All the sentences are encoded in utf-8 format.

2.2 Word Alignment

The first task is to align words from the parallel corpora between Hindi and the other languages. We have used IBM model 1 to 5 and HMM model to align the words using Giza++ (Och and Ney, 2000). Hindi shows a remarkable similarity with the other 4 Indo-Aryan languages considered for this work (Masica, 1993). With the other 3 Dravidian languages Hindi shares typological properties like word order, head directionality, parameters, etc (Krishnamurti, 2003). Being so similar in structure, these language pairs exhibit high alignment accuracies. The extracted translation

Table 1: Written scripts of various Indian languages

Language	Script
Bengali(Ben)	Bengali alphabet
Gujarati(Guj)	Gujarati alphabet
Hindi(Hin)	Devanagari
Konkani(Kon)	Devanagari
Malayalam(Mal)	Malayalam alphabet
Marathi(Mar)	Devanagari
Punjabi(Pun)	Gurmukhi
Tamil(Tam)	Tamil alphabet
Telugu(Tel)	Telugu alphabet
Urdu(Urd)	Arabic
English(Eng)	Latin (English alphabet)

pairs are then matched for phonetic similarity using LCS algorithm, as discussed in the following section.

2.3 Phonetic Matching

In the extracted translation pairs, we have to find whether these words are transliteration pairs or just translation pairs. The major issue in finding these pairs is that the languages are in different scripts and no distance matching algorithm can be applied directly. Using Roman as a common representation (Gupta et al., 2010), however, is not a solution either. A Roman representation will miss out issues like short vowel drop. For example, $ktAb$ (Urdu, book) and $kitAb$ (Hindi, book) (Figure 1), essentially same, are marked as non-transliteration pairs due to short vowel drop in Urdu (Kulkarni et al., 2012). We opt for a phoneme matching algorithm to bring all the languages into a single representation and then apply a distance matching algorithm to extract the transliteration pairs. Fortunately, such a scheme for Indian languages exists, which will be addressed in the 2.3.1.

2.3.1 Indic-Soundex

Soundex algorithm (Russell and Odell, 1918) developed for English is often used for phoneme matching. Soundex is an optimal algorithm when we just have to compare if two words in English sound same. Swathanthra Indian Language Computing Project (Silpa[5]) (Silpa, 2010) has proposed

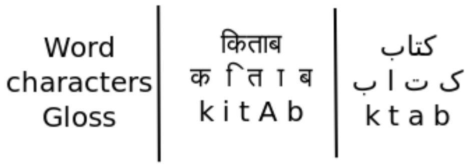


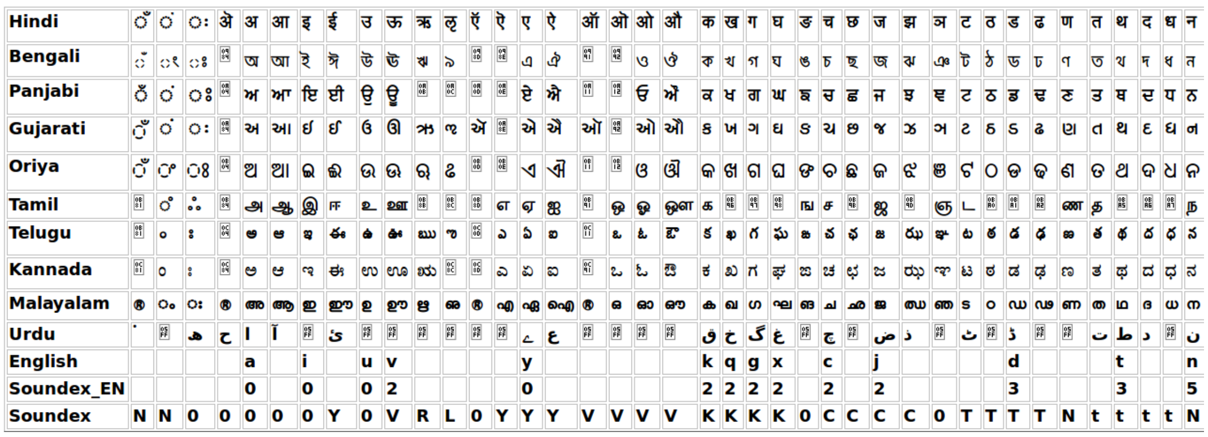
Figure 1: Figure shows $kitAb$ (book), written in Hindi and Urdu respectively, with their gloss (Hindi is written in Devanagari script from left to right while Urdu is written in Persio-Arabic script from right to left. The gloss is given from left to right in both). As is clear that if a both are transliterated into a common representation, they wont result into a transliteration pair

an Indic-Soundex system to map words phonetically in many Indian languages. Currently, mappings for Hindi, Bengali, Punjabi, Gujarati, Oriya, Tamil, Telugu, Kannada and Malayalam are handled in the Silpa system. Since Urdu is one of the languages we are working on, we introduced the mapping of its character set in the system. The task is convoluted, since with the other Indian languages, mapping direct Unicode is possible, but Urdu script being a derivative from Arabic modification of Persian script, has a completely different Unicode mapping[6] (BIS, 1991). Also there were some minor issues with Silpa system which we corrected. Figure 2 shows the various character mappings of languages to a common representation. Some of the differences from Silpa system include:

- mapping for long vowels.
  - U, o, au, are mapped to v.
  - E and ae are mapped to y.
  - A is mapped to a.
- $bindu$ and $chandrabindu$ in Hindi are mapped to n.
- ah and $halant$ are mapped to null (as they have no sound).
- Short vowels like a, e, u are mapped to null.
- h is mapped to null as it does not contribute much to the sound. It is just a emphasis marker.
- To make Silpa mappings readable every sound is mapped to its correspoding character in Roman.

[5]The Silpa Soundex description, algorithm and code can be found from http://thottingal.in/blog/2009/07/26/indicsoundex/. The Silpa character mapping can be found at http://thottingal.in/soundex/soundex.html

[6]Source: http://en.wikipedia.org/wiki/Indian_Script_Code_for_Information_Interchange



Hindi	◌ँ	◌ं	◌ः	ऄ	अ	आ	इ	ई	उ	ऊ	ऋ	ऌ	ऍ	ऎ	ए	ऐ	ऑ	ऒ	ओ	औ
Bengali	◌ঁ	◌ং	◌ঃ	□	অ	আ	ই	ঈ	উ	ঊ	ঋ	ঌ	□	□	এ	ঐ	□	□	ও	ঔ
Panjabi	◌ਁ	◌ਂ	◌ਃ	□	ਅ	ਆ	ਇ	ਈ	ਉ	ਊ	□	□	□	□	ਏ	ਐ	□	□	ਓ	ਔ
Gujarati	◌ઁ	◌ં	◌ઃ	□	અ	આ	ઇ	ઈ	ઉ	ઊ	ઋ	ઌ	ઍ	□	એ	ઐ	ઑ	□	ઓ	ઔ
Oriya	◌ଁ	◌ଂ	◌ଃ	□	ଅ	ଆ	ଇ	ଈ	ଉ	ଊ	ଋ	ଌ	□	□	ଏ	ଐ	□	□	ଓ	ଔ
Tamil	□	◌ஂ	ஃ	□	அ	ஆ	இ	ஈ	உ	ஊ	□	□	□	எ	ஏ	ஐ	□	ஒ	ஓ	ஔ
Telugu	□	◌ం	◌ః	□	అ	ఆ	ఇ	ఈ	ఉ	ఊ	ఋ	ఌ	□	ఎ	ఏ	ఐ	□	ఒ	ఓ	ఔ
Kannada	□	◌ಂ	◌ಃ	□	ಅ	ಆ	ಇ	ಈ	ಉ	ಊ	ಋ	□	□	ಎ	ಏ	ಐ	□	ಒ	ಓ	ಔ
Malayalam	®	◌ം	◌ഃ	®	അ	ആ	ഇ	ഈ	ഉ	ഊ	ഋ	ഌ	®	എ	ഏ	ഐ	®	ഒ	ഓ	ഔ
Urdu	·	□	ھ	ح	ا	آ	□	ئ	□	□	□	□	□	□	ے	ع	□	□	□	□
English					a		i		u	v					y					
Soundex_EN					0		0		0	2					0					
Soundex	N	N	0	0	0	0	0	Y	0	V	R	L	0	Y	Y	Y	V	V	V	V

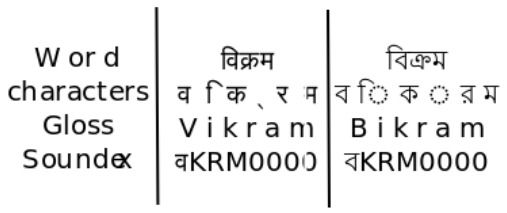
Hindi	क	ख	ग	घ	ङ	च	छ	ज	झ	ञ	ट	ठ	ड	ढ	ण	त	थ	द	ध	न
Bengali	ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ	ট	ঠ	ড	ঢ	ণ	ত	থ	দ	ধ	ন
Panjabi	ਕ	ਖ	ਗ	ਘ	ਙ	ਚ	ਛ	ਜ	ਝ	ਞ	ਟ	ਠ	ਡ	ਢ	ਣ	ਤ	ਥ	ਦ	ਧ	ਨ
Gujarati	ક	ખ	ગ	ઘ	ઙ	ચ	છ	જ	ઝ	ઞ	ટ	ઠ	ડ	ઢ	ણ	ત	થ	દ	ધ	ન
Oriya	କ	ଖ	ଗ	ଘ	ଙ	ଚ	ଛ	ଜ	ଝ	ଞ	ଟ	ଠ	ଡ	ଢ	ଣ	ତ	ଥ	ଦ	ଧ	ନ
Tamil	க	□	□	□	ங	ச	□	ஜ	□	ஞ	ட	□	□	□	ண	த	□	□	□	ந
Telugu	క	ఖ	గ	ఘ	ఙ	చ	ఛ	జ	ఝ	ఞ	ట	ఠ	డ	ఢ	ణ	త	థ	ద	ధ	న
Kannada	ಕ	ಖ	ಗ	ಘ	ಙ	ಚ	ಛ	ಜ	ಝ	ಞ	ಟ	ಠ	ಡ	ಢ	ಣ	ತ	ಥ	ದ	ಧ	ನ
Malayalam	ക	ഖ	ഗ	ഘ	ങ	ച	ഛ	ജ	ഝ	ഞ	ട	ഠ	ഡ	ഢ	ണ	ത	ഥ	ദ	ധ	ന
Urdu	ق	خ	گ	غ	□	چ	□	ض	ذ	□	ٹ	□	ڈ	□	□	ت	ط	د	□	ن
English	k	q	g	x		c		j					d				t			n
Soundex_EN	2	2	2	2		2		2					3				3			5
Soundex	K	K	K	K	0	C	C	C	C	0	T	T	T	T	N	t	t	t	t	N

Figure 2: A part of Indic-Soundex, mappings of various Indian characters to a common representation, English Soundex is also shown as a mapping. This is modified from one given by (Silpa, 2010)

In the following, we discuss the computation and extraction of phonetically similar word pairs using LCS algorithm.

2.3.2 Word-pair Similarity

Aligned word pairs, with soundex representation, are further checked for character level similarity. Strings with LCS distance 0 or 1 are considered as transliteration pairs. We also consider strings with distance 1 because there is a possibility that some sound sequence is a bit different from the other in two different languages. This window of 1 is permitted to allow extraction of pairs with slight variations. If pairs are not found to be exact match but at a difference of 1 from each other, they are checked if their translation probability (obtained after alignment of the corpora) is more than 50% (empirically chosen). If this condition is satisfied, the words are taken as transliteration pairs (Figure 3). This increases the recall of transliteration pair extraction reducing precision by a slight percentage. Table 2 presents the statistics of extracted transliteration pairs using LCS.



Word	विक्रम	বিক্রম
characters	व ि क ् र म	ব ি ক ্ র ম
Gloss	V i k r a m	B i k r a m
Soundex	वKRM0000	বKRM0000

Figure 3: Figure shows $Vikram$, a person name, written in Hindi and Bengali respectively, with their gloss and soundex code. The two forms are considered transliteration pairs if they have a high translation probability and don't differ by more than 1 character.

A detailed algorithm using translation probabilities obtained during alignment phase is provided in Algorithm 1.

Algorithm 1 match(w1, w2, translation-probability (w1,w2))

```
1. Find the language of both the words by using the 1st character of each and checking in the character list.
2. Calculate Soundex equivalent of w1 and w2 using Soundex algorithm.
3. Check if both the soundex codes are equal.
4. If yes, return both as transliteration pairs.
5. else, check the LCS between the soundex codes of w1 and w2.
6. If the distance is found to be 1,
7. check if the translation probability for w1 to w2 is more than 0.5.
8. if Yes, return both are transliteration pairs.
9. else both are not transliteration pairs.
```

2.4 Evaluation of Transliteration Pairs

To evaluate the phonetic similarity of the extracted aligned word pairs by LCS algorithm, a small subset (10%) from each language pair is given for Human evaluation. Annotators[7] are asked to judge whether the extracted word pairs are in fact transliterations of each other or not, based on the way the word pairs are pronounced in the respective languages. The results for the transliteration pairs for a given language pair extracted by

[7]Annotators were bi-literate graduates or undergraduate students, in the age of 20-24 with either Hindi or the transliterated language as their mother tongue

LCS algorithm are reported in Table 2. Hindi-Urdu and Hindi-Telugu (even though Hindi and Telugu do not belong to the same family of languages) demonstrate a remarkably high accuracy. Hindi-Bengali, Hindi-Punjabi, Hindi-Malayalam and Hindi-Gujarati have mild accuracies while Hindi-Tamil is the least accurate pair. Not only Tamil and Hindi do not belong to the same family, the lexical diffusion between these two languages is very less. For automatic evaluation of alignment quality we calculated the alignment entropy of all the transliteration pairs (Pervouchine et al., 2009). These have also been listed in Table 2. Tamil, Telugu and Urdu have a relatively high entropy indicating a low quality alignment.

Table 2: This table shows various language pairs with the number of word-pairs, accuracies of manually annotated pairs and the alignment entropy of all the languages. Here accu. represents average accuracy of the language-pairs

pair	#pairs	accu.	entropy
Hin-Ben	103706	0.84	0.44
Hin-Guj	107677	0.89	0.28
Hin-Mal	20143	0.86	0.55
Hin-Pun	23098	0.84	0.39
Hin-Tam	10741	0.68	0.73
Hin-Tel	45890	0.95	0.76
Hin-Urd	284932	0.91	0.79

3 Transliteration with Hindi as the Source Language

After the transliteration pairs are extracted and evaluated, we train transliteration systems for 7 languages with Hindi as the source language. In the following section, we explain the training of these transliteration systems in detail.

3.1 Creation of data-sets

All the extracted transliteration word pairs of a particular language pair are split into their corresponding characters to create a parallel data set, for building the transliteration system. The data-set of a given language pair is further split into training and development sets. 90% data is randomly selected for training and the remaining 10% is kept for development. Evaluation set is created separately because of two reasons; firstly we don't want to reduce the size of the training data by splitting the data set into training, testing and development, secondly evaluating the results on a gold data set would give us a clear picture of the performance of our system. Section 3.3 explans various evaluation methodologies for our transliteration systems.

3.2 Training of transliteration systems

We model transliteration as a translation problem, treating a word as a sentence and a character as a word using the aforementioned data-sets (Matthews, 2007, ch. 2,3) (Chinnakotla and Damani, 2009). We train machine transliteration systems with Hindi as a source language and others as target (all in different models), using Moses (Koehn et al., 2007). Giza++ (Och and Ney, 2000) is used for character-level alignments (Matthews, 2007, ch. 2,3). Phrase-based alignment model (Figure 4) (Koehn et al., 2003) is used with a trigram language model of the target side to train the transliteration models. Phrase translations probabilities are calculated employing the noisy channel model and Mert is used for minimum error-rate training to tune the model using the development data-set. Top 1 result is considered as the default output (Och, 2003; Bertoldi et al., 2009).

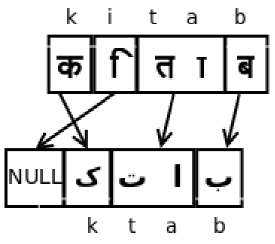


Figure 4: Figure depicts an example of phrase-based alignment on $kitAb$ (book), written in Hindi (top) and Urdu (bottom).

3.3 Evaluation

In this section we will do an in-depth evaluation of all the transliteration systems that we reported in this work.

3.3.1 Creation of Evaluation Sets

We used two data-sets for the evaluation. The creation of these data-sets is discussed below:

- **Gold test-set**: Nearly 25 sentences in Hindi, containing an approximate of 500 words (unique 260 words) were randomly extracted from a text and given to human annotators for preparing gold data. The annotators[7] were given full sentences rather than individual words, so that they could decide the correct transliteration according to the context. We were not able to create gold test-set for Tamil.

- **WordNet based test-set**: For automatic evaluation, the evaluation set is created from the synsets of Indian languages present in Hindi WordNet (Sinha et al., 2006). A Hindi word and its corresponding synsets in other languages (except Gujarati) are extracted and represented in a common format using Indic-Soundex and then among the synsets only exact match(s), if any, with the corresponding Hindi word, are picked. In this way, we ensure that the evaluation set is perfect. The set mainly contains cognate words (words of similar origin) and named entities.

3.3.2 Evaluation Metrics

We evaluated the transliteration systems on the above-discussed test-sets following the metrics discussed below:

- We used the evaluation metrics like ACC, Mean Fscore, MRR and MAP$_{ref}$, which refer to Word-accuracy in Top1, Fuzziness in Top1, Mean-reciprocal rank and precision in the n-best candidates repectively (Zhang et al., 2012).

- Keeping in view the actual goal of the task, we also evaluated the systems based on the readability of their top output (1-best) based on the transliteration of consonants. Consonants have a higher role in lexical access than vowels (New et al., 2008), if the consonants of a word are transliterated correctly, the word is most likely to be accessed and thus maintaining readability of the text. So, we evaluated the systems based on the transliteration of consonants.

3.3.3 Results

We present consolidated results in Table 3 and Table 4. Apart from standard metrics i.e., metrics 1, metrics 2 captures character-level and word-level accuracies considering the complete word and only the consonants of that word with the number of testing pairs for all the transliteration systems. The character-level accuracies are calculated according to the percentage of insertions and deletions required to convert a transliterated word to a gold word. Accuracy of all the transliteration systems is greater than 70%, i.e. even a worst transliteration system would return a string with 7 correct characters, out of 10, on an average. The accuracies at the character-level of only the consonants ranges from 75-95% which clearly proves our systems to be of good quality. It is clear from the results that these systems can be used as a reading aid by a non-Hindi reader.

As the table shows, all the transliteration systems have shown similar results on both the test-sets. These results clearly show that all the systems except Malayalam, Tamil and Telugu perform rather well. This can be attributed to the fact that these languages belong to the Dravidian family while Hindi is an Indo-Aryan language. Although, as per Metrics 1, the results are not promising for these languages, the consonant-based evaluation, i.e. Metrics 2, shows that the performance is not that bad.

Perfect match of the transliterated and gold word is considered for word-level accuracy. Bengali, Gujarati, Punjabi and Urdu yield the very high transliteration accuracy. The best system (Hindi-Pujabi) gives an accuracy of nearly 70% on word-level whereas Hindi-Urdu gives the highest accuracy on character-level. Urdu transliteration accuracy being so high is strengthened from the fact that linguistically the division between Hindi and Urdu is not well-founded (Masica, 1993, p. 27-30) (Rai, 2001). We can infer from the results of the word-level accuracies of the whole word that these transliteration systems cannot be directly used by a system for further processing.

3.3.4 Human Evaluation

In order to re-confirm the validity of the output in practical scenarios, we also performed human-based evaluation. For human evaluations 10 short Hindi sentences, with an average length of 10 words, were randomly selected. All these sentences were transliterated by all the 7 transliteration systems and the results of each were given to several evaluators [8] to rate the sentences on the scale of 0 to 4.

- *Score 0*: Non-Sense. If the sentence makes no sense to one at all.

[8] Annotators were bi-literate, some of who did not know how to read Hindi, graduates or undergraduate students, in the age of 20-24 with the transliterated language as their mother tongue

Table 3: Evaluation Metrics on Gold data.

Metrics 1					Metrics 2			
Lang	ACC[9]	Mean F-score	MRR	Map$_{ref}$	Char (all)	Char (consonant)	Word (consonant)	#Pairs
Ben	0.50	0.89	0.57	0.73	0.89	0.94	0.72	260
Guj	0.59	0.89	0.67	0.84	0.91	0.97	0.86	260
Mal	0.11	0.69	0.26	0.55	0.73	0.94	0.40	260
Pun	0.60	0.90	0.69	0.83	0.89	0.93	0.81	260
Tel	0.27	0.75	0.32	0.49	0.78	0.93	0.71	260
Urd	0.58	0.89	0.67	0.81	0.88	0.89	0.70	260

Table 4: Evaluation Metrics on Indo-WordNet data.

Metrics 1					Metrics 2			
Lang	ACC	Mean F-score	MRR	Map$_{ref}$	Char (all)	Char (consonant)	Word (consonant)	#Pairs
Ben	0.60	0.91	0.70	0.87	0.93	0.95	0.87	1263
Mal	0.15	0.78	0.31	0.61	0.83	0.88	0.71	198
Pun	0.69	0.92	0.76	0.88	0.70	0.73	0.47	1475
Tam	0.31	0.82	0.38	0.57	0.82	0.86	0.58	58
Tel	0.34	0.87	0.49	0.76	0.87	0.93	0.82	528
Urd	0.67	0.92	0.73	0.84	0.92	0.94	0.83	720

- *Score 1*: Some parts make sense but is not comprehensible over all.
- *Score 2*: Comprehensible but has quite few errors.
- *Score 3*: Comprehensible, containing an error or two.
- *Score 4*: Perfect. Contains minute errors, if any.

Table 5 contains the average scores given by evaluators for the outputs of various transliteration systems. The results clearly depict the ease that a reader faced while evaluating the sentences. According to these scores, Gujarati, Bengali and Telugu transliteration system gives nearly perfect outputs, followed by the transliteration systems of Urdu and Malayalam which can be directly used as a reading aid. Tamil and Punjabi transliterations were comprehensible but contained a considerable number of errors.

[9]ACC stands for Word level accuracy; Char(all) stands for Character level accuracy; Char(consonant) stands for Character level accuracy considering only the consonants; Word(consonant) stands for Word level accuracy considering only the consonants

Table 5: Average score (out of 4) by evaluators for various transliteration systems

language	avg. score
Bengali	3.6
Gujarati	3.8
Malayalam	3.3
Punjabi	1.9
Tamil	2.5
Telugu	3.6
Urdu	3.2

4 Conclusion

We have proposed a method for transliteration of Hindi into various other Indian languages as a reading aid for non-Hindi readers. We have chosen a complete statistical approach for the same and extracted training data automatically from parallel corpora. An adaptation of Soundex algorithm for a normalized language representation has been integrated with LCS algorithm to extract training transliteration pairs from the aligned language-pairs. All the transliteration systems return transliterations, good enough to understand the text, which is strengthened from the evaluators' score as well as from the character-level accuracies. However, word-level accuracies of these transliteration systems prompt them not to be used as a tool for text processing applications. Further, we are training transliteration models between all these 8 Indian languages.

References

Nicola Bertoldi, Barry Haddow, and Jean-Baptiste Fouet. 2009. Improved minimum error rate training in moses. *The Prague Bulletin of Mathematical Linguistics*, 91(1):7–16.

Bureau of Indian Standards BIS. 1991. *Indian Script Code for Information Interchange, ISCII*. IS 13194.

Sandeep Chaware and Srikantha Rao. 2011. Rule-Based Phonetic Matching Approach for Hindi and Marathi. *Computer Science & Engineering*, 1(3).

Manoj Kumar Chinnakotla and Om P Damani. 2009. Experiences with english-hindi, english-tamil and english-kannada transliteration tasks at news 2009. In *Proceedings of the 2009 Named Entities Workshop Shared Task on Transliteration*, pages 44–47. Association for Computational Linguistics.

Vishal Goyal and Gurpreet Singh Lehal. 2009. Hindi-punjabi machine transliteration system (for machine translation system). *George Ronchi Foundation Journal, Italy*, 64(1):2009.

Rohit Gupta, Pulkit Goyal, Allahabad IIIT, and Sapan Diwakar. 2010. Transliteration among indian languages using wx notation. *g Semantic Approaches in Natural Language Processing*, page 147.

Kanika Gupta, Monojit Choudhury, and Kalika Bali. 2012. Mining hindi-english transliteration pairs from online hindi lyrics. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, pages 23–25.

Jagadeesh Jagarlamudi and A Kumaran. 2008. Cross-Lingual Information Retrieval System for Indian Languages. In *Advances in Multilingual and Multimodal Information Retrieval*, pages 80–87. Springer.

Girish Nath Jha. 2010. The tdil program and the indian language corpora initiative (ilci). In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC 2010). European Language Resources Association (ELRA)*.

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

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.

Bhadriraju Krishnamurti. 2003. *The Dravidian Languages*. Cambridge University Press.

Amba Kulkarni, Rahmat Yousufzai, and Pervez Ahmed Azmi. 2012. Urdu-hindi-urdu machine translation: Some problems. *Health*, 666:99–1.

Jin-Shea Kuo and Ying-Kuei Yang. 2004. Generating paired transliterated-cognates using multiple pronunciation characteristics from Web Corpora. In *PACLIC*, volume 18, pages 275–282.

Gurpreet S Lehal and Tejinder S Saini. 2010. A hindi to urdu transliteration system. In *Proceedings of ICON-2010: 8th International Conference on Natural Language Processing, Kharagpur*.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, volume 10, page 707.

Abbas Malik, Laurent Besacier, Christian Boitet, and Pushpak Bhattacharyya. 2009. A hybrid model for urdu hindi transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 177–185. Association for Computational Linguistics.

Colin P Masica. 1993. *The Indo-Aryan Languages*. Cambridge University Press.

David Matthews. 2007. Machine transliteration of proper names. *Master's Thesis, University of Edinburgh, Edinburgh, United Kingdom*.

Animesh Nayan, B Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal, and Ratna Sanyal. 2008. Named entity recognition for Indian languages. *NER for South and South East Asian Languages*, page 97.

Boris New, Verónica Araújo, and Thierry Nazzi. 2008. Differential processing of consonants and vowels in lexical access through reading. *Psychological Science*, 19(12):1223–1227.

F. J. Och and H. Ney. 2000. Improved Statistical Alignment Models. pages 440–447, Hongkong, China, October.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Vincent Pagel, Kevin Lenzo, and Alan Black. 1998. Letter to sound rules for accented lexicon compression. *arXiv preprint cmp-lg/9808010*.

Kalyani Patel and Jyoti Pareek. 2009. Gh-map-rule based token mapping for translation between sibling language pair: Gujarati–hindi. In *Proceedings of International Conference on Natural Language Processing*.

Vladimir Pervouchine, Haizhou Li, and Bo Lin. 2009. Transliteration alignment. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 136–144. Association for Computational Linguistics.

Alok Rai. 2001. *Hindi nationalism*, volume 13. Orient Blackswan.

R Russell and M Odell. 1918. Soundex. *US Patent*, 1.

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2012. A statistical model for unsupervised and semi-supervised transliteration mining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 469–477. Association for Computational Linguistics.

K Saravanan and A Kumaran. 2008. Some experiments in mining named entity transliteration pairs from comparable corpora. *CLIA 2008*, page 26.

Silpa. 2010. Swathanthra Indian Language Computing Project. [Online].

Manish Sinha, Mahesh Reddy, and Pushpak Bhattacharyya. 2006. An approach towards construction and application of multilingual indo-wordnet. In *3rd Global Wordnet Conference (GWC 06), Jeju Island, Korea*.

Jörg Tiedemann. 1998. Extraction of translation equivalents from parallel corpora. In *Proceedings of the 11th Nordic conference on computational linguistics*, pages 120–128. Center för Sprogteknologi and Department of Genral and Applied Lingusitcs (IAAS), University of Copenhagen, Njalsgade 80, DK-2300 Copenhagen S, Denmark.

Muhammad Adeel Zahid, Naveed Iqbal Rao, and Adil Masood Siddiqui. 2010. English to Urdu transliteration: An application of Soundex algorithm. In *Information and Emerging Technologies (ICIET), 2010 International Conference on*, pages 1–5. IEEE.

Min Zhang, Xiangyu Duan, Vladimir Pervouchine, and Haizhou Li. 2010. Machine transliteration: Leveraging on third languages. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1444–1452. Association for Computational Linguistics.

Min Zhang, Haizhou Li, Ming Liu, and A Kumaran. 2012. Whitepaper of news 2012 shared task on machine transliteration. In *Proceedings of the 4th Named Entity Workshop*, pages 1–9. Association for Computational Linguistics.