Resource Creation for Training and Testing of Transliteration Systems for Indian Languages

Sowmya V.B.[*], Monojit Choudhury[*], Kalika Bali[*], Tirthankar Dasgupta[◊], Anupam Basu[◊]
[*]Microsoft Research Lab India, Bangalore, India
[◊]Society for Natural language Technology Research, Kolkata, India
E-mail: {t-sowmyv, monojitc, kalikab}@microsoft.com, {iamtirthankar, anupambas}@gmail.com

Abstract

Machine transliteration is used in a number of NLP applications ranging from machine translation and information retrieval to input mechanisms for non-roman scripts. Many popular Input Method Editors for Indian languages, like Baraha, Akshara, Quillpad etc, use back-transliteration as a mechanism to allow users to input text in a number of Indian language. The lack of a standard dataset to evaluate these systems makes it difficult to make any meaningful comparisons of their relative accuracies. In this paper, we describe the methodology for the creation of a dataset of ~2500 transliterated sentence pairs each in Bangla, Hindi and Telugu. The data was collected across three different modes from a total of 60 users. We believe that this dataset will prove useful not only for the evaluation and training of *back-transliteration* systems but also help in the linguistic analysis of the process of transliterating Indian languages from native scripts to Roman.

1. Introduction

Transliteration refers to the process of writing the text of one language using the script of another language whereby the sound of the text is preserved as far as possible (Knight and Graehl, 1998). Transliteration can be classified into two types: forward and backward. *Forward transliteration* refers to the process of representation of a word (in our context, Indian language word) using a non-native script (in this case, Roman script). For example, Roman string "sachin" might be generated by forward transliteration from the original Hindi word "सचिन" which is in the Devanagari script. *Back transliteration*, on the other hand, is the reverse process whereby one can obtain the native script representation back from the transliterated word. Thus, backward transliteration will generate the Devanagari string "सचिन" from the Roman string "sachin".

Automatic transliteration is useful in various NLP applications including monolingual and cross-lingual Information Retrieval and Machine Translation. Apart from these, back transliteration in particular can also be employed as a mechanism for text input especially for non-roman scripts. Transliteration as a mechanism for text input has also been discussed in (Sandeva *et al*, 2008) for Sinhalese and (Ehara and Kumiko, 2008) for multi lingual text entry. It has also been used for other applications like identifying cross-lingual spelling variations in names (Scott McCarley, 2009) and named entity recognition (Animesh *et al*, 2008). We observe that Roman transliterations of Indian language text are very common on the web especially in blogs, instant messaging and emails. The absence of standard keyboards for Indian languages, difficulty in learning existing keyboards, coupled with the familiarity with QWERTY keyboard, Roman script and English language for most of the Indian internet users, make the use of Roman transliterations of Indian languages fairly widespread.

While the study of transliteration of native words into Roman is linguistically interesting and useful in understanding the correspondence between the two scripts, this is also important for building forward and backward transliteration engines between Indian languages and English. Among other applications, a back-transliteration system from English to Indian languages can also be used as an Indian language input mechanism. In fact, to this end, there have been several back-transliteration systems for Indian languages. Some of them are used as desktop Input Method Editors (IMEs), like Baraha[1] while others are used as web applications, like Google Indic Transliterate[2] (currently supports 11 Indian languages), Quillpad[3] (currently supports 10 Indian languages). Microsoft Indic Language Input Tool[4] (currently available in 10 languages) provides both a desktop as well as a web-based version of transliteration based IME. All these systems follow different approaches to perform back-transliteration but without a standard dataset it is difficult to evaluate these systems on common grounds to make any meaningful comparisons.

The recently conducted NEWS workshop (Li *et al*, 2009) hosted a shared task of transliteration of named entities for eight language pairs, including three Indian languages namely Hindi, Tamil and Kannada. The dataset prepared for this task was restricted to named entities of various origins on either side, and therefore, is not exclusively designed and neither is it appropriate for

[1] www.baraha.com
[2] http://www.google.com/transliterate/indic
[3] http://quillpad.com/
[4] http://specials.msn.co.in/ilit/

training and evaluation of back transliteration systems, especially the Roman script based input mechanisms for Indian languages.

In this paper, we describe the creation and some initial analysis of a dataset of Indian language words transliterated into English words. Through various user experiments, we have created about 2500 pairs of transliterated sentences, totalling to approximately 25,000 words, in each of the three languages – Bangla, Hindi and Telugu. We believe that this dataset is useful for the linguistic study of the process of transliteration of native words from Indian languages to Roman script, and evaluation as well as training of back transliteration systems.

The rest of this paper is organized as follows: Section 2 explains our methodology for data collection and transcription. Section 3 presents an initial analysis of the data for all the three languages. Section 4 concludes the paper indicating future directions.

2. Methodology

We chose three languages – Bangla, Hindi and Telugu for the data collection process, primarily to study the effect of linguistic typology on the transliteration process. Bangla and Hindi belong to the Indo-Aryan family, while Telugu is a Dravidian language. Telugu is a highly agglutinative language, whereas Hindi is inflectional in nature. In terms of the extent of agglutination, Bangla falls somewhere in the middle. We also note that Hindi, Bangla and Telugu are amongst the largest Indian languages, having approximately 325 million, 196 million, and 74 million speakers respectively.[5]

The three sets of experiments conducted for each of these languages to collect transliteration data under natural settings are described in the following sections.

2.1 Mode of collection

The objective of the experiment was to collect natural Roman transliterations of Indian language sentences such as "anand shatranj tournament jeet liya" and then pair them up with the underlying original Hindi, Bangla and Telugu sentences, such as "*आनंद शतरंज टोर्नमेंट जीत लिया*" for Hindi. It was essential to obtain the most natural manner in which the user transliterated their language as we wanted to account for the variations across users as well as valid variations for the same user. A number of ascii-transliteration schemes such as ITRANS[6] and Baraha are available for Indian languages, and in the collection of this dataset it was necessary to ensure that the users did not follow any such pre-decided scheme. As described below, we have obtained the user transliterated data under *controlled* and *uncontrolled* settings.

In the *controlled* setting, the text that the user enters is decided *a priori* and user does not have control over the choice of vocabulary. This mode has been chosen to ensure language coverage, i.e., covering as many vowel and consonant combinations as possible. Data was collected by performing a *dictation* experiment, where users were given some speech files with Indian language sentences and were asked to listen and transcribe these sentences in Roman script. This process was adopted instead of "look and type" interface, to avoid the influence of the native spelling of the word that the visual presence of the original word might have on the transliteration. This ensured that the users used the transliteration scheme that came to them most naturally.

In the *uncontrolled* setting, the users were allowed to construct sentences of their own choice under two different modes: *scenario* and *chat*. While the scenario mode mimics blogging and emailing, the chat mode was designed to collect chat data. One major difference between these two modes is that while in the former the user has the luxury to read and edit his/her input, in the latter, the pressure to communicate in real time leaves no room for intensive editing.

In the *scenario writing* task, the users were asked to choose from a set of scenarios and write around 100 words each on any two of them using Roman script. The topics ranged from popular movies to current news items. In the *chat with the user* task, the users were asked to chat with a researcher, in their native language, using Roman script. These were general informal chat sessions on topics like the weather, the plan of the day, likes-dislikes of the users etc., which lasted for about ten minutes. Around 75 words per user were collected in this manner.

2.2 Dataset Preparation

For the *controlled setting* experiment, a set of 550 sentences were collected for each language from various sources ranging from news corpus to blogs and other web content. We ensured that the selected sentences covered as many of the valid letter-letter combinations for that particular language as possible. The chosen sentences were recorded by native speakers of the language. In all the three languages, every user was given 75 sentences for transcription. Of these, 50 sentences were common to all users and 25 were unique to a given user. This division was made such that the common sentences could be used for studying spelling variation patterns for a given word across individuals, while the unique sentences ensured coverage across the entire set of users.

2.3 User Selection

We have collected data from 18-20 users for each language. The users chosen were native speakers of the

[5] http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

[6] http://www.aczoom.com/itrans/

language, who use Roman script to type in Indian languages regularly for chatting, mailing etc. In India, Hindi is spoken and used quite frequently as a *lingua franca* by speakers of other Indian languages. Thus, for the Hindi experiments, in addition to native speakers, users with near-native like competence were chosen from other of regions and language groups of India.

2.4 Transcriptions

The collected user data needed to be back-transliterated into the respective Indian language scripts. This was done manually. The transcribers were instructed to mark instances of *code-mixing* and numerals. The transcribed unicode data was collected and aligned word by word with the users' Roman script data in a semi automatic process. This process involved checking the number of words in each sentence pair from users and transcribers. The cases of mismatch were understood as non-aligned and they were aligned manually by means of a simple user-interface.

3. Data Analysis

Table 1 describes the size of the dataset in terms of number of words.

We have collected various statistics on patterns of word and letter usage in the collected data. However, for paucity of space, here we report only two major observations: spelling variations in the Roman transliterations and the extent of code mixing, that is, the usage of English words within an Indian language text.

Mode of data collection	**Bangla**	**Hindi**	**Telugu**
Dictation (common)	6427	12934	13360
Dictation (unique)	4016	6592	6030
Scenario Data	3377	4044	4279
Chat Data	2648	2698	2276
Total	**16468**	**26268**	**25945**

Table 1: Size of the collected datasets (in words)

3.1 Spelling Variations

The data was analyzed for possible spelling variations during transliteration. We observed that for all the languages a large proportion of the words had only one spelling. However, there were also a number of words with a large number of variations.

Figure 1 compares the number of observed variations of a word (x-axis) plotted on a logarithmic scale with number of words found to have that many variations (y-axis). The trend is very similar across the three languages: we observe that most of the words exhibit very few spelling variations, whereas only very few words have a large number of variants (as large as 20) in the dataset. One reason behind this observation is the frequency distribution of the words themselves. A corollary of the Zipf's law says that frequency of a word is inversely proportionate to the number words having that frequency raised to certain positive power. In other words, there are few words with very high frequency and large number of words with one or two occurrences in the corpus. If we assume that the probability of observing a new spelling variation is almost fixed for every new occurrence of that word encountered in the corpus, then it implies that the number of variants of high frequency words will be large, whereas that of the low frequency words will be fewer. Since high (low) frequency words are rarer (abundant), so words with large (fewer) number of variations are also rarer (abundant).

Although this explanation holds to a good extent, apart from frequency, there are several other factors that determine the number of variants of a word. For instance, it is possible as well as typical to represent vowels and especially diphthongs using various Roman letter sequences. On the other hand, usually there are one-to-one mapping between the scripts for consonants. Therefore, the actual character sequence of a word also plays a significant role in determining the number of observed variants. This fact is illustrated in Table 2, which shows some sample words and their variations. Indeed, it clearly shows that two of the common reasons for spelling variations are:

- Ambiguity in vowel representation (like राज being written raja, raaja)
- Aspirated consonants (like ప్రభుత్వం being written as prabhutvam, prabutvam, prabhuthvam etc.)

These variations are not surprising as in the process of transliteration, a user is trying to map a large character set of Indian languages (more than 50 graphemes) to a relatively smaller set of English alphabet (26 letters). Further, certain conventions are region specific, for example, the aspiration in consonants in the Northern part of the country (or for speakers of Indo-Aryan languages) is represented by the addition of "h" to the consonant. Thus, the character for aspirated voiceless dental plosive "थ" in Hindi is mainly transliterated as "th". In the Southern part of the country (or for speakers of Dravidian languages), "h" is mainly used to indicate "dental" place of articulation, rather than aspiration. It is beyond the scope of this paper to go into the details of the linguistic basis of such conventions.

3.2 Code Mixing

Code-mixing, or the interspersing of English words in Indian language, is frequently observed in chat, blog and email texts. From the data, we studied the extent of code mixing across users in all the languages. Consider the Hindi sentence: यह क्रिकेट बॉल है (Translation: This is a cricket ball). A possible transliteration for this sentence is: "yaha kriket ball hai". In this example, यह and है are Indian origin words. On the other hand, क्रिकेट *(cricket)* and बॉल (ball) are of English origin. Therefore, somebody could potentially type in the English spellings of these words instead of transliterating them. In the example at hand, clearly kriket has been transliterated, whereas in ball, the original spelling has

been retained. We consider the latter as a *genuine* case of code-mixing, while both of them as a *potential* case of code mixing.

Figure 2 and Figure 3 show the cumulative distribution across users of the percentage of *genuine* code-mixing and ratio of *genuine* to *potential* cases of code-mixing respectively. The x-axes of the plots show the percentage of code-mixing, while the y-axes show the number of subjects who were observed to have a code-mixing propensity smaller than or equal to a specific percentage. From Figure 2 we observe that all the users for Bangla, Hindi and Telugu have respectively exhibited *at most* 8%, 11% and 12% genuine code-mixing. However, only very few subjects actually show this high propensity. A lot of users (13 for Bangla, 15 for Hindi and 16 for Telugu) actually show less than 6% genuine code-mixing.

Figure 3 has to be interpreted in a similar fashion, except for the fact that here x-axis represent the percentage of genuine-to-potential cases of code-mixing. It is interesting to note that around 10 users for Hindi and 2 for Telugu had 100% genuine-to-potential code-mixing. This means that lot of users for these languages type the actual English spelling whenever there is a scope for doing so.

From this analysis of code-mixing across languages, we have made the following observations:

1. Chat data had more cases of genuine code mixing compared to scenario data – across all languages. This can imply that people tend to perform more code mixing during conversations than otherwise.
2. The extent of genuine code-mixing across users have a similar trend for all the languages, though on an average, Telugu and Bangla users had more code-mixing compared to Hindi users.
3. On the other hand, the ratio of genuine to potential code-mixing is less than 50% for a considerable number of Bangla users. This indicates that there is a high tendency for Bangla users to type in non-English sound-based spellings for English words.

4. Conclusion

In this paper, we have described the creation of a dataset for Indian language transliteration data for Bangla, Hindi and Telugu, which we believe will be useful in developing and evaluation of Roman script based input mechanisms for Indian languages, and also, in general, Indian language to English transliteration systems. We believe that our methodology though designed for Indian languages, is generally applicable to the collection of any transliterated data where it is essential to obtain data in natural form to ensure coverage and user-variation. It is possible that the methodology may prove useful for other domains of data-collection, like spoken language data and transcriptions.

An initial analysis of the data collected across the languages indicates that there are specific linguistic and socio-linguistic phenomena that need to be dealt with to account for variation across users. It is essential for any transliteration based system, especially IMEs, to take care of variations in spellings for higher accuracies and wider applicability.

Currently we are collecting similar data for two more Indian languages – Tamil and Kannada. We are also studying the characteristics of the dataset and working on understanding the effects of various features on the user data.

5. References

Ehara, Y.,Tanaka-Ishii, K. (2008). Multilingual text entry using automatic language detection. In *Proceedings of IJCNLP 2008*, pp. 441 – 448.

Goonetilleke, S., Hayashi, Y., Itoh,Y., Kishino, F. (2008). SriShell Primo: A Predictive Sinhala Text Input System. In *Proceedings of IJCNLP 2008 workshop on NLP for less privileged languages*. pp. 43 – 50.

Knight, K. and Graehl J. (1998). Machine Transliteration. *Computational Linguistics*, 24(4), pp. 599 – 612.

Li, H., Kumaran, A., Zhang, M., Pervouchine, V. (2009). Report of NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop*. pp. 1 – 18.

McCarley, J.S. (2009), Cross language name matching. In *Proceedings of ACM-SIGIR 2009*, pp. 660 – 661.

Nayan, A., Rao, B.R.K., Singh, P., Sanyal, S., Sanyal, R. (2008). Named Entity Recognition for Indian Languages. In *Proceedings of IJCNLP 2008 workshop on NER for South and South-East Asian languages*. pp. 97 – 103.

Word	Pronunciation (in IPA)	Variations	Example Variations
Bangla			
নিজস্ব	nidʒoʃʃo	14	nijosho;nijoso;nijoshwa;nijoswo;nijoshsho;nijoshyo;nijoswa;
ধনাঢ্য	dʰɔnaddʰo	17	dhonadhoi;dhanadhya;dhonyadhyo;dhonadyo;dhannaddho;
অভ্যেস	obbʰeʃ	20	ovesh;obbhes;abhyash;abhyesh;abhyes;abhyas;avyas;obbhesh;
সহ্য	ʃodʒdʒʰo	13	sojho;sahya;sojhyo;sojjho;sojhjho;sajhya;shojjo;sajya;
হওয়ার	hɔoar	11	hoyar;haoar;hoar;hayoar;houar;howar;hoyoar;
Hindi			
राजस्व	radʒəsvə	16	rajasva;raajasav; rajaswa;rajsva; rajsv;raajasva;raajaswa;
बढ़ोतरी	bəɽʰotrI	15	badhotri; bhadhotri;badotri;badhottree;badothri;badhottari;
गाँव	gãv	13	gaaw;gaon;gav;gaanv;gaaon; gaav;gaun;gaao;
इकाइयों	ikaijõ	12	ikaayio;ikaaiyon;ekaiyon;ekaion;ikaiyon; ikaiyo;ikayiyo;ikayiyon
Telugu			
ప్రభుత్వం	prəbʰutvəm	8	prabhutvam;prabhuthvam;prabhuthavam;prbhutvam; prabutvam;
నిర్ణయించారు	nirɳəiɲʧaru	7	nirnayinchaaru;nirnyincharu;nirnayinchaaaru;nirnayimcharu;
ఉన్నాయి	unnaji	10	unnayi;unnai;vunnayi;vunnavi;unnay;
అయితే	aite	10	ayithe;aite;ite;ayite;ayyite;aithe;aaite;
తరువాత	təruvata	9	taruvaata;taaruvaata;tarvata;taruvata;taruvatha;

Table 2: Spelling variations across languages

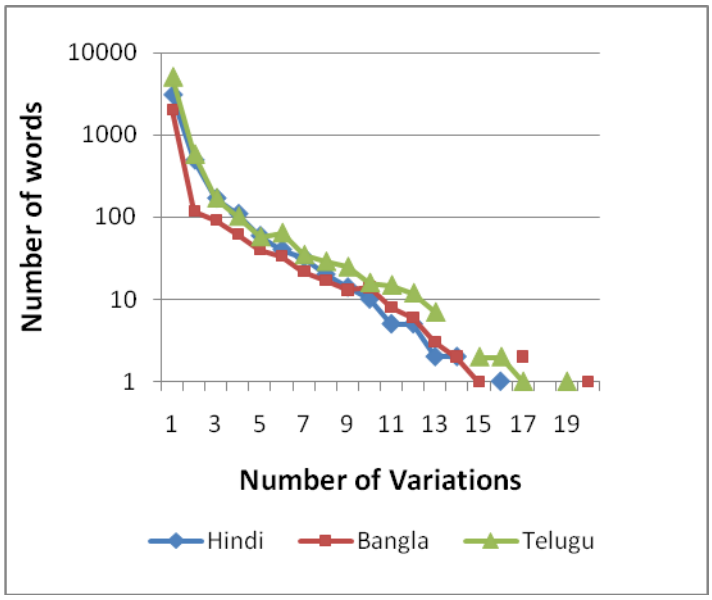

Figure 1: Number of words Vs Number of variations

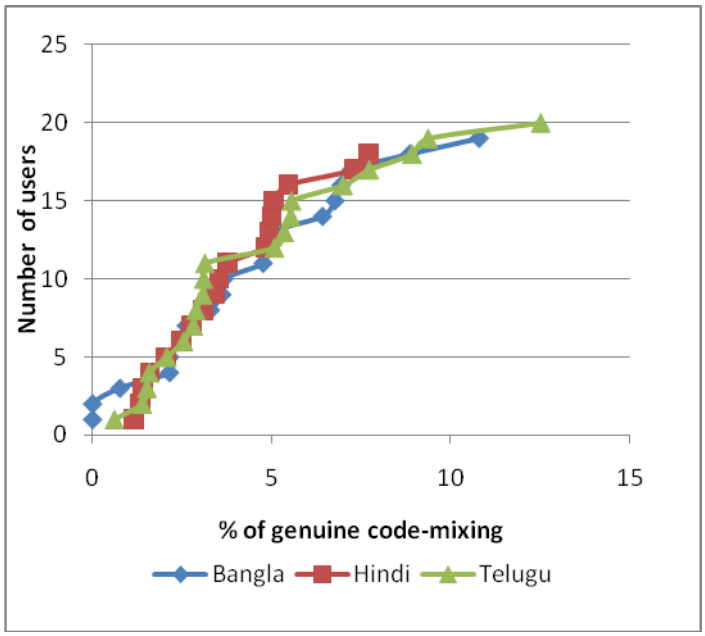

Figure 2: Cumulative Distribution of the % of genuine code-mixing across users

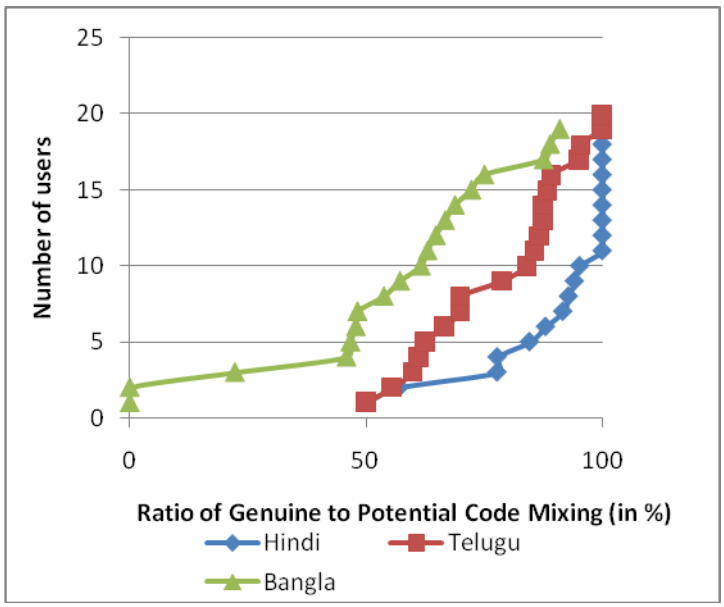

Figure 3: Cumulative Distribution of the ratio of genuine to potential code-mixing