Antaryāmī: The Smart Keyboard for Indian Languages

Amitava Das
Department of Computer Science
University of North Texas
amitava.das@unt.edu

Abstract

The evolution of the World Wide Web and mobile devices has presently created a giant number of textual communications. The life of a modern person is now saturated with textual information and interactions such as SMS(s), e-mails, chats, social media texts and so on. Therefore the need continuously rises for smart input methods. Smart keyboards, enabled with next word prediction and auto completion capabilities have became very popular for English already, but there is no such effort for Indian languages. Here this paper reports on the development process of a smart keyboard called antaryāmī for two of the most popular Indian languages Hindi and Bengali. An Indian word "antaryāmī" means the one who knows the innermost thoughts and feelings. Likewise the developed smart keyboard provides next word prediction and auto completion facilities in a de facto manner as if it knows the user's intentions and thoughts, thus the keyboard has named as antaryāmī / अंतरजामी/अन्तर्यामी.

1 Introduction

Text is still the primary medium for all kind of communication and with the advent of WWW and smart-mobile devices like cell phones, tabs, and etc. have added another dimension to the textual information interchange in the form of SMS(s), e-mails, chats, social media texts: Facebook, Twitter messages and so on. To bring up with more user friendliness there are various smart keyboard apps for English are been built such as SwiftKey[1], Fleksy[2] and so on and constantly evolving. These apps help users by giving multiple suggestions for the next possible word while typing. For example the keyboard interface as shown in the fig 1 predicting the next possible word for the given input text sequence.

There is no such tool available for Indian languages so far. Therefore Romanization based phonetic input became the benchmark for Indian languages. Perhaps the best-known phonetic input based keyboard for Indian language is the Quillpad[3], supports 10 languages. There are a few tools available for specific languages. For example Avro Keyboard[4] for Bengali is a widely used one, supports phonetic input based Unicode conversion. Even the Google transliteration[5] platform supports the same phonetic typing mechanism. A tool of this kind gives multiple alternatives as a dropdown list while typing. Fig 2 shows an example of Quillpad for Hindi phonetic typing.

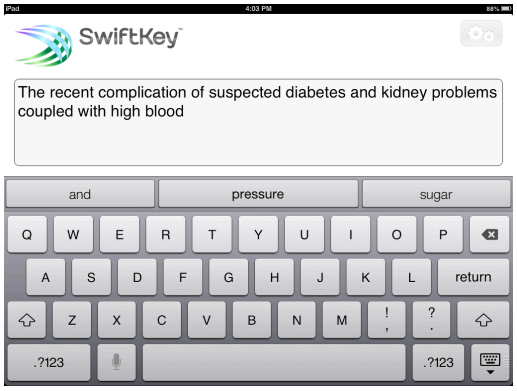

Figure 1: Smart Keyboard for English

Figure 2: Phonetic Typing for Hindi

While all the above systems for Indian languages are popular and using automatic transliteration strategies for input there is still not enough scientific study on the English-to-Indian languages transliteration in the context of general purpose typing (Salam et. al., 2012). So far transliteration studies on these languages are

1 http://www.swiftkey.net/en/
2 http://fleksy.com/
3 http://www.quillpad.in/
4 http://www.omicronlab.com/avro-keyboard.html
5 http://www.google.com/inputtools/cloud/try/

concentrated on named entities only. Indeed more research efforts required on this domain.

On the contrary there is no Indian languages keyboard available, which can predict next word on the fly while typing and can autocorrect wrongly typed words. Even state-of-the-art systems for English like SwiftKey, and Fleksy enable users to write whole phrases even without lifting their finger. So, the motivation for the present paper is to develop a smart keyboard solution for Indian languages both for PC and mobile devices.

The developed antaryāmī keyboard is capable to phonetically transliterate the Romanized word inputs to Unicode and can also predict next possible input word, smart enough to correct typing errors and can learn user writing style.

2 Related Works

Research on Input Method Editors (IME) has received researcher's interest recently. A dedicated workshop called Workshop on Advances in Text Input Methods (WTIM)[6] on this issue held twice in 2011 and 2012. In these workshops a few research attempts have been reported on IME development for Indian languages. Those endeavors include languages like Hindi, Bengali (Salam et. al., 2012), Telugu (Ahmed et. al., 2011), Assamese (Saharia and Konwar, 2012) and Bramhi (Brouillette et. al., 2012) and non-Indian languages like Arabic (Chalabi and Gerges, 2012) and Dzongkha (Dasgupta et. al., 2012) and so on. All these works targeted on phonetic typing and back transliteration. The motivation of the present paper is entirely different, that is to develop a smart adaptive software keyboard for Indian languages, which can predict next input word, can fix typing errors and finally can automatically learn user writing style.

Research paper on smart keyboard development is not very common. Might be the reason is that it is a very new topic and another important reason is that smart keyboard is one of the hot revenue generating business ideas presently; therefore none of the manufacturer is ready to make their architecture open for research.

3 Corpus

Facebook interactions written in Indian languages have been collected for the present task. Phonetic Romanization is the standard practice for all the Indian languages and the use of Unicode is very infrequent. Indeed unavailability of smart IMEs one of the prime reason of Romanization but even large numbers of users prefer English alphabets for easy typing. A typical Romanized Indian language Facebook text looks like the following example:

Yaar tusi great ho!

Unicode: इयार तुसी great हो!

Eng.: My friend you are great!

Even beside the phonetic typing and Romanization code-mixing is a standard practice in Indian sub-continent. According to the Twitter language map[7], Europe and South-East Asia are the most language-diverse areas currently exhibiting high Twitter usage. Code-mixing is very frequent in these regions, since languages change over a very short geospatial distance and people generally have basic knowledge of the neighboring languages. Here this paper concentrates on India, a nation with some 1600 spoken languages, of which 30 have more than 1 million speakers, while 22 are official languages. Both language diversity and dialect changes instigate frequent code-mixing in the country. Indians are multilingual by adaptation and frequently change and mix languages in textual interactions. Language identification from the code-mixed text is a separate ongoing task where the same data is being used.

Code-mixing has 3 major categories such as **inter**- vs **intra**-sentential mixing (depending on whether it occurs outside or inside sentence or clause boundaries); **intra**-word vs **tag** switching (if the mixing occurs within a word, e.g., at a morpheme boundary, or by inserting a tag phrase or word from one language into another), and on whether the mixing is an act of identity in a group or if it is competence-related (i.e., a consequence of a lack of competence in one of the languages).

The corpus has all these types of code-mixing. Only monolingual and intra-word or tag mixing cases are been kept rest of the code-mixing types i.e. inter/intra-sentential portion of the corpus are excluded for the present task. Altogether 4200 Hindi sentences and 10288 Bengali sentences are finally extracted. A simple dictionary based semi-automatic method has been adopted for the code-mixing type detection. Moreover it has been found that on a more formal topic people

6 http://research.microsoft.com/en-us/events/wtim2/

7http://www.flickr.com/photos/walkingsf/6277163176/in/photostream/

use less code-mixing compared to informal topic. The details of that system are omitted here due to space limitation.

Therefore an IME for Indian languages has few more challenges than English. Those additional challenges are proper general domain transliteration, language identification from code-mixed text along with the common qualitative challenges of next word prediction and auto completion.

Here the system has been made for two languages Hindi and Bengali. Hindi and Bengali are the two largest languages in India in terms of first language speakers, and 4th and 6th worldwide, respectively. Various campus Facebook groups were used for data acquisition, as detailed in Table 1. For future analysis two types of topics have been chosen including both formal and informal. Informal data are generally on fun topics such as on-campus love confession, on-campus matrimonial, etc. whereas formal topics include on campus technological forum, placement group and etc.

	Facebook Group	Type
Bengali	JU Confession	Fun
	JU Matrimonial	Fun
	Placement 2013 Batch	Formal
Hindi	IITB Confession	Fun
	IITB Compliments	Fun
	Tech@IITB	Formal

Table 1: Details of Corpus Sources

For the Bengali, the data came from Jadavpur University, which is located in Eastern India where the native language of most of the students is Bengali. For Hindi, the data came from the Indian Institute of Technology Bombay (IITB), an institution located in the west of India where Hindi is the most common language. An argument could be raised that choosing a formal corpus like news or travel could be a good alternative, but this data gives a real picture of everyday conversation and therefore it has been anticipated that the developed smart keyboard on this data would be more realistic in the context of mobile devices. Table 2 presents the corpus statistics for both the languages.

Here in the table the whole collected corpus statistics has been reported though a portion of that corpus is been used in the present task. Beyond the scope of the smart keyboard development this corpus is a valuable resource by itself. It could be used for language detection from code-mixed text, psycholinguistic study and even for various other linguistic studies as well.

Corpus Size		
Number Of Sentences	HND	8901
	BNG	24216
Number of Words	HND	67402
	BNG	193367
Number of Unique Token	HND	40240
	BNG	100227

Table 2: Corpus Size Statistics

4 Part-of-Speech (POS) Tagging

Syntactic information is very much useful for next word prediction. For example if someone has typed:

```
I (PRP) will (VAUX)
```

then possibly the next word will be a verb. An automatic word prediction of this kind is achievable by a POS based language model. There is no POS tagger available for the data (social media text (SMT)) type that has been chosen for the present task. SMT is noisy, in particular is characterised by having a high percentage of spelling errors and creative spellings (*gr8* for *'great'*), phonetic typing, word play (*goooood* for *'good'*), abbreviations (*OMG* for *'Oh my God!'*), Meta tags (*URL, Hashtags*), and so on.

General purpose POS taggers[8] are available for formal grammatically correct Hindi and Bengali text have been used to initiate. The tool uses 26 tagset[9] but which is not really relevant for the SMT, therefore a CMU POS tagset specially designed for the SMT (twitter) has been adopted for the present task. The details of that tagset for SMT has been described in the (Gimpel et al., 2011). Accordingly a tagset-mapping table has been prepared.

Due to lack of domain specific annotated data a bootstrapping method has been chosen for the POS tagging system development. Initially 500 sentences for both the languages have been tagged by the general purpose POS taggers and then the output tags are been converted to the CMU POS tagset automatically. These sentences then manually edited to fix all the errors. 300 out of 500 sentences then used as a seed list and a CRF[10] based classifier has been trained on this dataset. Rest 200 sentences are used as a test set. Now in each iteration 100 random unlabelled sentences have been chosen to tag by the system.

8http://ltrc.iiit.ac.in/showfile.php?filename=downloads/shallow_parser.php
9http://ltrc.iiit.ac.in/tr031/posguidelines.pdf
10http://crfpp.googlecode.com/svn/trunk/doc/index.html?source=navbar

Those sentences are then manually corrected and added with the original training set and the CRF system is then retrained with the augmented training data. During the manual correction system's performance has been measured after eaach iteration. Iterations are repeated until the performance of the system reaches a satisfactory level. Performance of the final system measured on the test set of 200 sentences is found 85.43% (F-Measure) for Hindi and 88.83% (F-Measure) for Bengali. Altogether approximate 15 iterations needed to reach the final accuracy level. For the annotation purpose Sanchay[11] annotation tool has been used.

5 Phonetic Transliteration

The antaryāmī interface has an optional phonetic transliteration service. The Modified Joint Source-Channnel model as described in (Das et. al., 2010) has been used for the English-to-Indian languages back transliteration. The transliteration system is originally trained on named entity corpus and therefore its performance on general purpose transliteration is not very good. Back transliteration from romanized input to Indian languages is a separate challenging problem and will be addressed next.

6 Word Predictions

Word predictions and their accuracies are the main essence of smart keyboard experiences. Two types of word predictions are generally involved in this kind of tools. The first one is half-typed word prediction. A smart keyboard gives ranked options for each half-typed word. For example if only "*g*" has been typed, then options will be given for the most probable words such as "*great*", "*good*", and so on. The second type of prediction is contextual next word prediction. An appropriate example has been shown in fig 1. The antaryāmī interface gives 3 alternatives for both the prediction types. The development processes of the word prediction modules of the antaryāmī system are been discussed below.

The background of the both types of predictions are based on POS n-gram and word n-gram, therefore those lists are pre-computed and stored accordingly. Smart keyboard experiences need real time processing thus hashmap, regular expression and others low complexity mechanism have been chosen for any kind of sorting and searching.

6.1 Next Word Prediction

Next word prediction accuracy is very much important for smart keyboard experiences. This particular prediction came out on the screen when someone press the space key i.e. end of a word input. To make the automatic predictions the system makes use of word n-gram list. There are various trade-offs when deciding on the value of n for word n-grams. The alphabet of English is limited to 26 characters, and Romanization was used for the other two languages, so the set of possible character n-grams remain manageable for small values of n. We carried out experiments for n = {1, 2, 3, 4} and found 3-gram and 4-gram to be the best optimum choices. While the n-gram list has been generated a normalized weight has been assigned with each entry. The weight has been calculated using the formula 1. Where w_n^i is the calculated weight of the i^{th} n-gram, c_n^i is the frequency of the particular n-gram in the total corpus and N is the total number of unique n-grams in the corpus.

$$w_n^i = c_n^i / N \quad ----(1)$$

No stemming has been used in the present setup. Speed is one of the key aspect of smart keyboard experiences and processing power of the mobile devices are very much limited, therefore it has been decided to not use any time taking NLP operations. Hopefully good corpus size and user modeling (discussed in section 7) are the best alternatives of stemming for the particular problem domain. An example of the antaryāmī interface with next word prediction options is reported in the following fig 3.

Figure 3: Antaryāmī Next Word Prediction

11 http://sanchay.co.in/

The antaryāmī interface gives multiple ranked alternatives for the next possible word. A simple POS n-gram and word n-gram based heuristics have been used to produce the ranked suggestions.

- POS 4-gram list has been used to predict the next possible POS sequence for a given input sequence. A list of possible POS categories is generated and the list is sorted according to their possibilities, which is calculated based on the n-gram co-occurrence frequencies.
- If the number of candidates in the generated possible POS category list from the POS 4-gram is lesser than 3 then it goes for another iteration with the POS 3-gram list.
- If the number of candidates in the generated possible POS category list from the POS 3-gram is lesser than 3 then it goes for another iteration with the POS 2-gram list.
- Now the system has 3 most probable POS categories. Lets say they are POS^1, POS^2 and POS^3 and their weights are $freq^1$, $freq^2$ and $freq^3$. The weights are basically the frequencies extracted from the n-gram list.
- Now a possible word list has been generated based on the word n-gram frequencies. The list has n-gram frequencies attached with each candidate and those scores are been treated as weights further. The list then sorted in descending order.
- The word n-gram list then filtered based on the decided POS categories. It means if any word from the word list fall into the 3 POS categories is been kept or excluded otherwise.
- Weights for each list members are been calculated by simple multiplication of POS frequency and word n-gram frequency.
- The list finally sorted based on the associated weights and top 3 candidates are shown as best alternatives.

For the initials positions the system has to rely on bigrams (for the 2nd word) or trigrams (for the 3rd word) instead of POS and word 4-grams. From the evaluation result it could be easily concluded that the longer anyone type the prediction accuracies will increases.

6.2 Half-Typed Word Prediction

Half-typed word prediction situation only occurs when a user doesn't get appropriate word in the next word prediction. The possible word list generated after the fifth iteration during the next word prediction is used here. It has been hypothesized that the appropriate word might be extracted but got lower ranking in the previous step. Therefore a new re-ranking methodology has been designed for this stage.

- At first only those words are been kept which are starting with the half-typed characters rest of the words are excluded.
- Now those chosen words are again re-sorted according to their frequency weights in a descending order.
- Top 3 words are chosen as best alternatives and shown as suggestion.

Figure 4: Antaryāmī Half-Typed Next Word Prediction

6.3 Auto Completion

There are two situations when auto complete has to take place. First when user press the space button after a half-typed word then the system has to pick the most probable first word from the available alternatives. Since hitting the spacebar instigates next word prediction by default always, therefore auto-complete requirement checking has been done. Simply it has been checked if the last typed word is one of the last 3 predicted words, then it has been decided as a correct input otherwise the system choose the auto-correction alternative. The second situation occurs when user typed something wrong that does not has any "*startwith*" match with the alternatives.

As said for the first situation simply the first alternative has been chosen. For the second situation a Minimum Edit Distance based measure has been calculated for choosing the right candidate among the alternatives.

7 User Modeling

Everybody else has different writing style. There are numerous works on automatic author profiling including automatic age, gender detection (Argamon et. al., 2009), and personality detection and on stylometry study.

The antaryāmī system is capable to learn user-writing style automatically. The longer the system will be used the prediction accuracies will increase. A very simple mechanism has been adopted for the user modeling. Different user POS n-gram and word n-gram lists have been maintained along with the default n-gram tables. While a new entry has been added in those lists a higher weightage has been assigned to the user-produced n-grams.

As a result while the system generate next predictions the user specific n-grams will come up among the best 3 choices.

8 Evaluation

For evaluation, the antaryāmī system was deployed to smartphones and iPads. Four users took part in the evaluation. Out of the four users two were native Hindi speakers and the other two native Bengali speakers. Participants used the keyboard for two months. The evaluation version of the antaryāmī had all the built in mechanisms so that it can store all the data for evaluation purposes. Users were requested to use antaryāmī for all their outgoing text interactions such as SMS(s), Facebook, Twitter and WhatsApp messages.

Three main aspects have been tested during the evaluation: *typing speed, prediction qualities, and the auto-completion accuracies*.

8.1 Typing Speed

Before going into the evaluation of typing speed, let's discuss the standards of typing speed on smartphones. There is no research paper available on this issue so far, but some competitions have been held on it and that can be treated as a benchmark.

In June 2011 Guinness World Records committee organized a typing contest using the QWERTY mobile phone[12]. Grace Pak (USA) typed a prescribed 264-character text on a QWERTY mobile phone in 56.57, an average of 4.66 characters per second. The text was:

> "The telephone was invented by Alexander Graham Bell (UK), who filed his patent for the telephone on 14 February 1876 at the New York Patent Office, USA. The first intelligible call occurred in March 1876 in Boston, Massachusetts, when Bell phoned his assistant in a nearby room and said 'Come here Watson, I want you."

The startup company Fleksy from California asked a few users with gold-medal-worthy typing skills to type the same text has been used in the Guinness World Record competition, but using their product[13]. They found that the top 10 winners were all unofficially faster than the official Guinness World Record for the required text. The first place typist was 25 percent faster than the world record, having finished the phrase in just 16.49 seconds, i.e., on average 16 characters per second. More than 100 users achieved gold status with the app – meaning they finished the phrase in 30 seconds or less. So 16 characters per second is thereby considered as the benchmark. But this study is for English and there is no similar study for Indian languages.

For the typing speed evaluation all the outgoing texts from those four devices have been recorded along with associated timestamps. Here the timestamp refers to the total typing time for any text i.e. the difference between the time when typing started and the time when it has been sent. The typing speed of antaryāmī users for both the languages has been recorded as being 9-10 characters per second, which is roughly double the speed of world record, but still little far from the Fleksy system word rates. The reason behind this might be the complexities of Indian languages due to consonant clusters/juktakkhors and difficult orthographies using matras. Indian languages also have single grapheme to multiple phoneme mapping ambiguities. Therefore a user can write different Romanized spelling for a single word and the system then becomes unable to predict the half-typed word. For example, the Hindi word "kya/kiya" (**Eng.** What) could be written in two ways, and if the system is trained with only one of these forms, then ambiguities will occur.

12http://www.guinnessworldrecords.com/records-4000/fastest-typing-on-a-smartphone/

13http://fleksy.com/2013/08/28/fleksy-users-shatter-world-record-in-smartphone-typing-competition/

This is an ongoing work and there are two anticipated ways-out for these problems. One is to increase relevant training corpus and the second one is to learn user modeling automatically.

8.2 Prediction Qualities

Prediction qualities for typed-in Indian languages words have been measured in various ways, including half-typed word prediction and next word prediction, and accuracies have been measured with word positions. It has been noticed that for both the prediction types accuracies are poorer for initial positions, as the system has less context for prediction, but for later positions, the system is performing well. Another trend is that the accuracies do not change much after a certain word position is reached. Prediction accuracies for both the types are changing very less among word positions (3-6) and (6-9). Therefore it is eminent that making automatic prediction for initial words is more challenging. Results are reported in the following Table 3.

Here is Table 3; both the prediction types results are presented in percentage format for simplicity, but that percentage for each of the perdition type has different implication. For the next word prediction, percentages refer to the cases, when users find an appropriate choice from the suggested list. More specifically when a space has been pressed to the typed-in text the system provide 3 possible alternatives for the next word and this percentage denote a ratio between: number of times user select a word from the automated suggested alternatives and the total number of words typed by the user over the evaluation period. There are few exceptional cases: if a user select an alternative from the automated suggestion list but then modify it, has not been considered as a right prediction. The percentage measure for the half-typed word prediction is a ratio between: required number of character input to get the right prediction and the total number of characters in that particular word. For example suppose someone has to type the word "*sambhalke*" (**Eng.** carefully) and s/he has to type 4 characters i.e. "*samb*" to get the right prediction, therefore the calculated prediction accuracy will be (4/9) = 44%. It could be described in another way that the antaryāmī system can save 66% typing effort. *Lastly for both the languages Hindi and Bengali prediction accuracies are more or less same, thus results are reported in the Table 3 in general not language specific*.

Prediction Types	Word Positions		
	1-3	3-6	6-9
Half-typed word	46%	65%	68%
Next Word	52%	70%	71%
With User Modeling			
Half-typed word	52%	75%	82%
Next Word	68%	80%	85%

Table 3: Word Prediction Accuracies

A separate evaluation has been done on user modeling. It has been mentioned that the evaluation has been done on two months of data. Though user modeling is a parallel process, but to understand the significance of user modeling a separate evaluation has been done. During the first month of the evaluation no user log data has been used for any kind of predictions but the system keep collecting the data in the background. After the first month the user log data has been added separately and been used for both types of predictions. From the result it is clear that the user modeling has significant performance increment over the normal baseline predictions.

No evaluation has been done for the transliteration module. Phonetic transliteration is an optional service in the antaryāmī interface and user can switch it on based on their preference, but in the present evaluation setup the evaluators are been asked to not include the service. Back transliteration from Romanized input to Indian languages is a separate challenging problem and will be addressed next.

8.3 Auto-Completion Accuracies

Basic auto-complete prediction accuracy is a fine-grained measure over the half-typed word prediction. It has been mentioned earlier (section 6.3) that the auto-complete function always picks the first candidate from the next word prediction list therefore auto-complete accuracy is a strict measure on system's accuracy over the half-typed word prediction.

Auto-complete function has another category to take care off i.e. typing mistake. For both the categories results are reported in the Table 4.

Auto-completion accuracies are indeed not very good and user modeling did not help much in this case. Different strategies have to be taken for this kind of problem. Might be a language specific orthographic spelling correction (UzZaman and Khan, 2004) technique is a better alternative for this problem kind. This section is

now under research. Detailed discussion with the evaluators revealed that due to the bad performance of the auto-completion they carefully avoided this option.

Auto-Completion Types	Word Positions		
	1-3	3-6	6-9
Half-typed word	26%	28%	29%
Typing Mistake	18%	20%	20%
With User Modeling			
Half-typed word	26%	35%	35%
Typing Mistake	18%	22%	23%

Table 4: Auto-Completion Accuracies

9 Conclusion

Development of smart IME(s) for various languages is a very new topic and there are very less number of published works in this domain. The main reason might be that the smart keyboard development companies want to keep their business ideas protected. But with the advances of mobile devices an adaptive software keyboard with word predictions and spell correction facilities has become an important mainstream NLP research topic.

This paper reports an initial study on smart adaptive software keyboard development for the Indian languages, but still there is a long way to go and more research efforts are indeed needed. What has been realized from this development experience is that the technology has to be more fast to provide the real time predictions because the mobile devices have space and processing limitations.

Proposed algorithms in this paper have been reported as a step-by-step process with the philosophy that anyone can replicate the architecture easily for their own languages.

Finally to conclude it should be mentioned that this is an ongoing work. More challenges are there due to code-mixing, back transliteration and real time word prediction. Those issues are being considered presently.

References

Ahmed, U.Z.; Bali, K.; Choudhury, M.; and VB, S. (2011). Challenges in Designing Input Method Editors for Indian Languages: The Role of Word-Origin and Context. In the Proceeding of the First Workshop on Advances in Text Input Methods (WTIM 2011), IJCNLP 2011, pages 1-9.

Argamon, S.; Koppel, M.; Pennebaker, J.; and Schler J. (2009). Automatically profiling the author of an anonymous text, Communications of the ACM 52 (2): 119–123.

Brouillette, A.; Sarmah, D.; and Kalita J. (2012). Multi-objective Optimization for Efficient Brahmic Keyboards. In the Proceeding of the Second Workshop on Advances in Text Input Methods (WTIM 2), COLING 2012, pages 29-44.

Chalabi, A.; and Gerges, H. (2012). Romanized Arabic Transliteration. In the Proceeding of the Second Workshop on Advances in Text Input Methods (WTIM 2), COLING 2012, pages 89-96.

Das, A.; Saikh, T.; Mondal, T.; Ekbal, A.; and Bandyopadhyay, S. (2010). English to Indian Languages Machine Transliteration System at NEWS 2010. In the Proceeding of the named Entity Workshop (NEWS 2010), ACL 2010, pages 1-11.

Dasgupta, T.; Sinha, M.; and Basu, A. (2012). Forward Transliteration of Dzongkha Text to Braille. In the Proceeding of the Second Workshop on Advances in Text Input Methods (WTIM 2), COLING 2012, pages 97-106.

Gimpel, K.; Schneider, N.; O'Connor, B.; Das, D.; Mills, D.; Eisenstein, J.; Heilman, M.; Yogatama, D.; Flanigan, J.; and Smith, N.. (2011). Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In the Proceedings of ACL 2011, pages 42-47.

Saharia, N.; and Konwar, K. M. (2012). LUITPAD: A Fully Unicode Compatible Assamese Writing Software. In the Proceeding of the Second Workshop on Advances in Text Input Methods (WTIM 2), COLING 2012, pages 79-88.

Salam, K.; Yamada, S.; and Tetsuro, N. (2012). Phonetic Bengali Input Method for Computer and Mobile Devices. In the Proceeding of the Second Workshop on Advances in Text Input Methods (WTIM 2), COLING 2012, pages 73-78.

UzZaman, N.; and Khan, M. (2004). A Bangla Phonetic Encoding for Better Spelling Suggestions. In the Proceeding of 7th International Conference on Computer and Information Technology (ICCIT 2004),