Development of a Complete Urdu-Hindi Transliteration System

Gurpreet Singh LEHAL[1] *Tejinder Singh SAINI*[2]
(1) Department of Computer Science, Punjabi University, Patiala
(2) ACTDPL, Punjabi University, Patiala
gslehal@gmail.com, tej74i@gmail.com

ABSTRACT

Hindi and Urdu are variants of the same language, but while Hindi is written in the Devnagri script from left to right, Urdu is written in a script derived from a Persian modification of Arabic script written from right to left. The difference in the two scripts has created a script wedge as majority of Urdu speaking people in Pakistan cannot read Devnagri, and similarly the majority of Hindi speaking people in India cannot comprehend Urdu script. To break this script barrier, it becomes necessary to develop a high accuracy Urdu-Devnagri transliteration system. The major challenges in developing such system are handling missing diacritic marks and short vowels in Urdu, zero/multiple character mappings of Urdu in Hindi, absence of half characters in Urdu, multiple mappings of Urdu words in Hindi and word segmentation issues in Urdu including broken and merged words. Already a few Urdu-Hindi transliteration systems have developed but their accuracy is not very high and they have failed to address all the above issues. For the first time, we present a complete Urdu-Hindi transliteration system which takes care of all the above issues and has reported a transliteration accuracy of more than 97% at word level.

KEYWORDS : Urdu, Hindi, Devnagri, Machine Transliteration, Language Models

Proceedings of COLING 2012: Posters, pages 643–652,
COLING 2012, Mumbai, December 2012.

1 Introduction

Hindi and Urdu are variants of the same language, but while Hindi is written in the Devnagri script from left to right, Urdu is written in a script derived from a Persian modification of Arabic script written from right to left. Hindi is the official language of India, while Urdu is the national language of Pakistan, and also one of the state languages in India. The spoken form of the two languages is very similar. Since Urdu and Hindi are grammatically same language and they also share a very good number of words, it is easier for both speakers to understand each others' language. The only obstacle is the script. Thus there is an urgent need to develop a high accuracy Urdu-Devnagri transliteration system. Already some work in this direction has been reported (Malik at el. 2008, Malik at el. 2009), but these systems suffer from low accuracy and have not handled some of the major transliteration issues such as resolving word ambiguity. Some work has also been reported on the reverse Hindi-Urdu transliteration (Bushra and Tafseer, 2009; Duranni et al., 2010; Lehal and Saini, 2010; Sajjad et al., 2011; Visweswariah et al., 2010).

In the following sections, we shall be discussing the major challenges in developing a high accuracy Urdu-Hindi transliteration system. The linguistics and language models along with the algorithms developed to meet these challenges are also discussed in detail, followed by experimental results. When there is no confusion, we use the terms Devnagri and Hindi interchangeably.

2 Challenges in Urdu-Hindi Transliteration

The major challenges of transliteration of Urdu to Hindi are as follows:

- **Recognition of Urdu Text without Diacritical Marks:** Diacritical marks are sparingly used in Urdu, even though they are critical for correct pronunciation and disambiguation of certain words. These missing diacritical marks create substantial difficulties for transliteration systems.
- **Filling the Missing Script Maps:** There are many characters which are present in the Urdu script, corresponding to those having no character in Devnagri, e.g. Hamza ء , Do-Zabar ً, Aen ع, Khadi Zabar ٖetc.
- **Multiple mappings for Urdu characters:** It is observed that corresponding to many Urdu characters there are multiple mappings into Devnagri script (example و -> व, ो, ौ, ु, ू, ऊ, ओ, औ). Grammar rules and context are needed to select the appropriate Devnagri character for such Urdu characters.
- **Transliteration ambiguity at word level:** There are many Urdu words which map to multiple Hindi words. For example: میل (मेल, मील, मैल) / بچے (बचे, बच्चे) / کیا (क्या, किया) / ہوا (हुआ, हवा). Higher level language information will be needed to choose the most relevant Hindi word.
- **Word-Segmentation Issues:** Space is not consistently used in Urdu words, which makes word segmentation a non-trivial task. Many times the space is deleted resulting in many Urdu words being jumbled together and many other times extra space is put in word resulting in over segmentation of that word. These words can still be easily understood by Urdu readers, but complicate the transliteration task.

- **Compound words in Urdu:** There are many compound words or combinations of Urdu words written as a multi-word expression in Hindi. For example: نقش قدم (नक्श-ए-क़दम), جوش و خروش (जोश-ओ-ख़रोश).

3 Our Approach

3.1 Lexical Resources Used

In order to perform statistical analysis during the various phases of the transliteration system we have developed lexical resources from Urdu and Hindi Corpora. The resources include a parallel corpus of Urdu-Hindi words/compound words/phrases, Urdu word based unigram language model, a statistical trigram character model for Hindi Language and Hindi word based unigram, bigram and trigram language models.

3.2 System Architecture

The system architecture of the Urdu-Hindi transliteration system is shown in Figure 1. The complete Urdu-Hindi transliteration system is divided into three stages: pre-processing, processing and post-processing. The three stages are discussed in detail in the following sections.

3.3 Pre-processing

In the pre-processing stage, the Urdu words are cleaned and prepared for transliteration by normalizing the Urdu words as well as joining the broken Urdu words. The two main stages in pre-processing are:

3.3.1 Normalizing Urdu words

There are a few Urdu characters that have multiple equivalent Unicodes. As for example, from transliteration point of view, ى(0649), ي(064a) and ى(06cc) represent the same Urdu character, similarly آ (0622) can be also be represented by the combination ا (0627)+ ◌ٓ (0653). All such forms are normalized to have only one representation.

3.3.2 Joining the broken Urdu words

The Urdu-Hindi transliteration system faces many problems related to word segmentation of Urdu script, as in many cases space is not properly put between Urdu words. Sometimes it is deleted resulting in many Urdu words being jumbled together and many other times extra space is put in word resulting in over segmentation of that word. The Urdu text can still be easily read by the reader, but when such words are transliterated to Hindi they produce erroneous results. So it is necessary to handle such space related errors. The space insertion problem is handled in both pre-processing and post-processing stage, while the space deletion problem is handled in the processing stage. The space insertion problem usually occurs due to conventional way of writing in Urdu or due to extra space being inserted during typing. The typing related space insertion problems are handled by using the algorithm suggested by Lehal (Lehal, 2009) in the pre-processing.

3.4 Processing Stage

In this stage, corresponding to each Urdu word, one or several possible Hindi words are generated. For multiple alternatives, the final decision is taken in the post processing stage. In the first pass, the Urdu sentence is parsed word by word and the Urdu word combinations are replaced with equivalent Hindi word combinations in the source Urdu sentence.

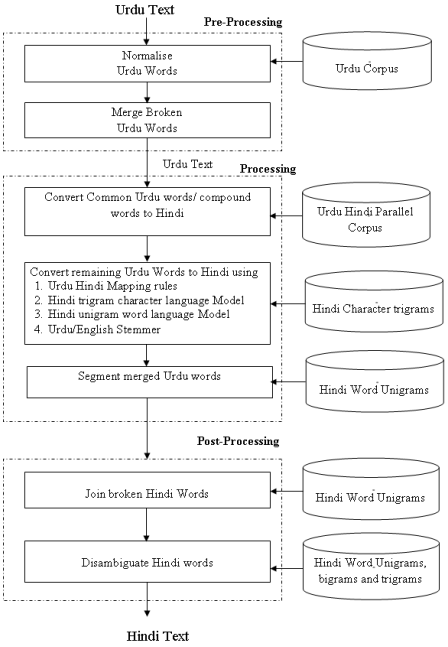

FIGURE 1 –System Architecture

In the next pass, the remaining Urdu words in the sentence, which could not be transliterated to Hindi in the first pass are processed. A multi-stage transliteration engine has been developed, to convert each Urdu. The Urdu words passes through each stage, till it gets converted to a non-empty set of Hindi words. The four stages are:

Stage 1: This stage uses a Language-model-based-generator(LMG) to convert the Urdu word. The LMG uses Urdu-Hindi character rules and a trigram character language model to generate a set of Hindi words from the Urdu word. A unigram word language model is then used to rank these words. If there is no word with non-zero probability, then we move to next stage.

Stage 2: In this stage, an attempt is made to extract the root form of the Urdu word using stemming rules for Urdu and English. If the root is found, then LMG is used to generate set of Hindi words corresponding to the Urdu root word. If root cannot be extracted or LMG returns an empty set, then we go to the next stage.

Stage 3: In this stage, the Urdu word is inspected for presence of merged words which can be transliterated to non empty sets of Hindi words. If no such sets can be generated then the word is sent to the next stage.

Stage 4: An Urdu word reaches this stage, if it cannot be transliterated to Hindi in the above 3 stages. In practice, very few words (only 0.39%) reach this stage. This stage uses the simple character mapping rules to convert the word to Hindi.

3.4.1 Language-model-based-generator (LMG)

This is the major module in the transliteration engine. It generates multiple Hindi transliterations for an Urdu word. The sequence of probable Hindi words is produced by a hybrid system based on rule based character mapping table between Urdu and Hindi characters and a trigram character Language Model. The Urdu word is processed character by character and the Urdu characters are mapped directly to their corresponding similar sounding Hindi characters (snippet shown in the Table 1).

Unicode	Urdu	Hindi
0627	ا	अ, ा
062a+06be	تھ	थ
0646	ن	न, ण, ं, ँ
0648	و	व, ो, ौ, ु, ू, ऊ, ओ, औ
06cc	ی	य, ी, े, ै, इ, ई, ए, ऐ

TABLE 1– Portion of Urdu-Hindi character mapping Table

In most of the cases, there is a 1-1 mapping from Urdu to Hindi characters. But there are a few characters such as و ,ن ,ی, which have multiple mappings. Special rules have been written to handle such cases, for example for character ی(o6cc):

- if ی(o6cc) is preceded by ء (0621), then replace both characters by ई
- if ی(o6cc) is followed by ا (0627), then replace ی by य
- if ی(o6cc) is preceded by ا (0627), then replace ی by ए
- if ی(o6cc) is preceded by a consonant, then replace ی by ी

More than 100 such rules have been written. But it was found that these rules have proved successful only in producing crude transliteration (54.19% accuracy) which is refined in subsequent stages. The first refinement we made was by using a character-based trigram language model, to decide between the various possible alternatives, as shown in Table 1. As a result, the transliteration accuracy increased to 72.62%, but it is still much behind the desired goal. The major reason was this poor accuracy was that in Urdu there are no half characters and also the diacritical marks are usually omitted in written text. The challenge is to fill these missing diacritical marks and put half characters at appropriate locations in Hindi word.

To solve this problem, for each Urdu character, we consider all its possible mappings in Hindi which include the missing short vowels and half characters. As an example, the Urdu character د, which we had mapped only to Devnagri character द, could practically map to द, दु, दि, द् , द्द, द्दु and द्दि (دلدل -> दलदल, دنیا -> दुनिया, دل -> दिल, گدگد -> गद्गद्, مدت -> मुद्दत, تشدد -> तशद्दुद, مجدد -> मुजद्दिद).

So we modify our mapping table to include all such forms for all the Urdu consonants resulting in multiple mappings. As a result, the 66 Urdu characters are mapped to 302 Hindi character combinations. We form all possible combinations, which could be generated from these multiple mappings and the top N combinations are retained. The character based trigram language model for Hindi is used to select the top N combinations. Each character combination may contain upto 4 Hindi characters. As for example, the Urdu character د (062F) can be mapped to Hindi character combination द्दि (0926+094D+0926+093F). These 302 possible mappings in Hindi, lead to 302^3 = 18,514,412 possible character trigrams.

As an example, consider the Urdu word کینسر(cancer). The five characters in the word : ک(क कि कु क् क्क क्कि क्कु), ی (य य् य्य य्यि य्यु यि यु ि ी े ै), ن (न न् न्न नि नु ण् ण णु णि ं ँ), س (स स् स्स सु सि स्सि स्सु) and ر (र र् रि रु र् रुि रुि) have multiple mappings in Hindi, as shown in brackets. Hence a total of 7*11*11*7*7 = 41,503 transliteration candidates have to be considered (Example: कैंसर किन्नसर केणसुर्र केन्सर etc.). After removing the combinations containing zero probability trigrams, the top five suggestions output by the trigram character language model are: केंसर, केंसरि, कींसर, कैंसर, कींसरु. We can see that not all the suggestions generated by the character language model are valid Hindi words. To further rank these words, we use the Unigram Word Model for Hindi. Words with non-zero probability in the Unigram word model are ranked based on their probabilities. In above example, we found that only two words had probability greater than zero and the ranked sequence of words produced by Unigram word model is: कैंसर, केंसर. It could also happen that all the top N alternatives suggested by the character level trigram may be having zero probabilities, in which case no alternative is returned by LMG.

3.4.2 Urdu/English Stemmer

It often happens that the root word maybe present in Hindi corpus but its inflections may not be present in Hindi corpus. In that case, the unigram language model will give zero probability for any such inflection and the word will not be considered for transliteration. To take care of situations, we use a light weight stemmer to obtain the root word and then transliterate it. A novelty in our stemmer has been that besides Urdu words, we also cover the English words which are also now frequently being used in Urdu.

3.4.3 Merged word segmentation

As already discussed above, space is not consistently used in Urdu, which gives rise to both space omission and space insertion errors. Due to the space deletion problem, a sequence of words is jumbled together as a single word and when the LMG tries to generate the equivalent Hindi alternatives it fails. The sequence of Urdu words written together without space is still readable because of the character joining property in Urdu. As for example, consider the word cluster انکارکردیاہے , which is composed of four words دیا , کر ,انکار and ہے . The Urdu readers can very easily segment and read the four words separately, but the computer will read them as a single word since there is no space in between, the LMG module fails to produce valid Hindi word. So it becomes necessary to break the jumbled word into individual words. We have used the space deletion algorithm presented by Lehal (Lehal, 2010) to split the Urdu words and then transliterate them.

For out of vocabulary words, the Hindi word is generated by using the trigram character language model and the top alternative is selected for further processing.

3.5 Post Processing Stage

Two main tasks are performed in the post processing stage. The first task is to join the broken words in Hindi and the second and more important task is to choose the best alternative, where ever multiple alternatives for Hindi words exist. The broken words are joined using the algorithm suggested by Lehal (Lehal, 2009). To choose between the different Hindi word alternatives we have used the word trigram probability. To take care of the sparseness in the trigram model, we have used deleted interpolation, which offers the solution of backing away from low count trigrams by augmenting the estimate using bigram and unigram counts. The deleted interpolation

trigram model assigns a probability to each trigram which is the linear interpolation of the trigram, bigram, unigram and uniform models as follows:

$$\Pr(w_i \mid w_{i-2}w_{i-1}) = \lambda_3 \frac{c(w_{i-2}w_{i-1}w_i)}{c(w_{i-2}w_{i-1})} + \lambda_2 \frac{c(w_{i-1}w_i)}{c(w_{i-1})} + \lambda \frac{c(w_i)}{\mathrm{N}} + \lambda_0 \frac{1}{V}$$

Where N = Number of words in the training corpus, V = Size of the vocabulary. The weights are set automatically using the Expectation-Maximization (EM) algorithm.

4 Results and Examples

We show with an example, the various stages of our Urdu-Hindi transliteration system in Table 2. The multiple transliteration options generated by the system are shown in braces.

Source Urdu Sentence: اسٹوڈینٹس نے گیارہ مارچ کو پریس میں کہا کہ موسیٰ احمد بے قصور ہے اور دہشت کا دور ختم ہو
After Pass1 : Searching Parallel Corpus कि मूसा اسٹوڈینٹس نے گیارہ مارچ کو پریس میں کہا احمد بے قصور ہے اور دہشت کا دور ختم ہو
After Stage 1 : LMG اسٹوڈینٹس ने ग्यारह मार्च کو پریس < में, मैं > कहा कि मूसा < अहमद, एहमद > बे < क़सूर, क़ुसूर > < है, हे > < और, ओर, अवर > < दहशत, दहश्त > का < दूर, दौर, दवर, दोर > < ख़त्म, ख़तम> < हो, ह, हौ >
After Stage 2 : Urdu/English Stemmer स्टूडैंटस ने ग्यारह मार्च کو پریس < में, मैं > कहा कि मूसा < अहमद, एहमद > बे < क़सूर, क़ुसूर > < है, हे > < और, ओर, अवर > < दहशत, दहश्त > का < दूर, दौर, दवर, दोर > < ख़त्म, ख़तम> < हो, ह, हौ >
After Stage 3 : Splitting merged words स्टूडैंटस ने ग्यारह मार्च < को, कौ, कू > <प्रेस, प्रैस > < में, मैं > कहा कि मूसा < अहमद, एहमद > बे < क़सूर, क़ुसूर > < है, हे > < और, ओर, अवर > < दहशत, दहश्त > का < दूर, दौर, दवर, दोर > < ख़त्म, ख़तम> < हो, ह, हौ
Post Processing Stage 1 : After Joining broken Hindi words स्टूडैंटस ने ग्यारह मार्च < को, कौ, कू > <प्रेस, प्रैस > < में, मैं > कहा कि मूसा < अहमद, एहमद> बेक़सूर < है, हे > < और, ओर, अवर > < दहशत, दहश्त > का < दूर, दौर, दवर, दोर > <ख़त्म, ख़तम> < हो, ह, हौ >
Post Processing Stage 2 : After selecting the best alternative स्टूडैंटस ने ग्यारह मार्च को प्रैस में कहा कि मूसा अहमद बेक़सूर है और दहशत का दौर ख़त्म हो

TABLE 2–Various transliteration stages

We compare our transliteration output with other available online systems. The transliteration/translation produced by these systems is shown in Table 3. The wrong translations and transliterations are marked in red colour.

5 Experimental Results

We have tested our system on 45 pages of Urdu Unicode text compiled from three Urdu websites. The text contained 18403 words. The transliterated text has been manually evaluated. The results are tabulated in Table 4. We can see how the transliteration accuracy increases in each stage with the addition of new language models and other linguistic resources. The initial transliteration accuracy obtained when the text was transliterated using the simple rule based character mapping

is 54.19%. The accuracy improved to 72.62% after application of trigram character language model. Later when the word based unigram language model was applied on the N words returned by the trigram character language model to select the word with highest unigram probability, the accuracy further improved to 82.58%. On combining the parallel Urdu-Hindi corpus, the accuracy increased to 92.81%. A further improvement in the accuracy was observed when the Urdu/English stemmer and word segmentation routines were added and the accuracy went upto 95.24%. And finally on applying the Hindi trigram word language model the accuracy reached 97.74%.

Urdu Sentence: اسٹوڈینٹس نے گیارہ مارچ کو پریس میں کہا کہ موسٰی احمد بے قصور ہے اور دہشت کا دور ختم ہو
Transliterated Hindi Sentence by Puran (http://www.sanlp.org/humt/HUMT.aspx) असटोडेंटस ने गयारा मारच कोपरेस में कहा कि मोसाय अहमद बे क़सोर हे ओर दहशत का दोर ख़तम हो
Translated Hindi Sentence by Sampark (http://www.tdil-dc.in/components/com_mtsystem/CommonUI/homeMT.php) असटोडीनटस ने ग्यारह मार्च कोपरीस में कहा ख मोस?ई अहमद बे अपराध है और आतंक का समय समाप्त हो
Translated Hindi Sentence by Google Translation (http://translate.google.com) ास्टोडीनटस ने ग्यारह मार्च कोपरेस कहा कह मूसा अहमद निर्दोष है और आतंक का युग समाप्त हो
Transliteration by our system (http://uh.learnpunjabi.org) स्टूडैंटस ने ग्यारह मार्च को प्रैस में कहा कि मूसा अहमद बेक़सूर है और दहशत का दौर ख़त्म हो

TABLE 3– Transliteration/Translation by some of the existing systems

6 Conclusion

In this research paper we have presented an Urdu to Hindi transliteration system which has achieved an accuracy of 97.74% at word level. The various challenges such as multiple/zero character mappings, missing diacritic marks in Urdu, multiple Hindi words mapped to an Urdu word, word segmentation issues in Urdu text etc. have been handled by generating special rules and using various lexical resources such as Hindi character trigram model, Hindi unigram, bigram and trigram word models, Urdu unigram model, Urdu-Hindi parallel corpus etc.

Linguistic/Statistical Resources Used	Transliteration Accuracy
Character based Mapping	54.19 %
Hindi Trigram Character Language Model	72.62 %
Hindi Word Based Unigram Language Model	82.58 %
Parallel Urdu-Hindi Corpus	92.81 %
Urdu/English Stemmer	92.93 %
Word Segmentation	95.24 %
Hindi Trigram Word Language Model	97.74%

TABLE 4 – Transliteration Accuracy in different stages

Acknowledgement

The authors would like to acknowledge the support provided by ISIF grants for carrying out this research.

References

Bushra J., Tafseer A. (2009). Hindi to Urdu Conversion: Beyond Simple Transliteration. In *Proceedings of the Conference on Language & Technology*, pages 24-31, Lahore, Pakistan.

Durrani, N., Sajjad, H., Fraser, A. and Schmid, H. (2010). Hindi-to-Urdu machine translation through transliteration. In *Proceedings of the 48th Annual Conference of the Association for Computational Linguistics*, pages 465–474, Uppsala, Sweden.

Lehal, G. S. and Saini, T. S. (2010). A Hindi to Urdu Transliteration System. In *Proceedings of 8th International Conference on Natural Language Processing*, pages 235-240, Kharagpur, India.

Lehal, G. S. (2009). A Two Stage Word Segmentation System For Handling Space Insertion Problem In Urdu Script, In *Proceedings of World Academy of Science, Engineering and Technology*, volume 60, pages 321-324, Bangkok, Thailand.

Lehal, G. S. (2010) A Word Segmentation System for Handling Space Omission Problem in Urdu Script. In *Proceedings of the 1st Workshop on South and Southeast Asian Natural Language Processing (WSSANLP)*, pages 43–50, the 23rd International Conference on Computational Linguistics (COLING), Beijing.

Malik, A., Boitet, C. and Bhattacharyya, P. (2008). Hindi Urdu machine transliteration using finite-state transducers. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 537–544, Manchester, UK.

Malik, A., Besacier, L., Boitet, C. and Bhattacharyya, P. (2009). A hybrid model for Urdu Hindi transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 177–185, Singapore.

Sajjad, H., Durrani N., Schmid, H. and Fraser, A. (2011). Comparing Two Techniques for Learning Transliteration Models Using a Parallel Corpus. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP2011)*, pages 129-137, Chiang Mai, Thailand.

Visweswariah, K., Chenthamarakshan, V. and Kambhatla, N., (2010) Urdu and Hindi: Translation and sharing of linguistic resources In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*: Poster Volume, pages 1283–1291, Beijing.