Transliterated Search using Syllabification Approach

Hardik Joshi[1] , Apurva Bhatt[1], Honey Patel[2]

[1]Department of Computer Science, Gujarat University, Ahmedabad, India.
{hardikjjoshi, apurva.bhatt7}@gmail.com
[2]L.J. College of Engineering, Ahmedabad, India
honeypatel.39@gmail.com

Abstract. Machine transliteration refers to the process of automatic conversion of a word from one language to another without losing its phonological characteristics. In this work, we present our experiments performed in subtask-1 and subtask-2 as a part of the FIRE-2013 transliterated search task. In both the subtasks, the transliteration from Roman script to Devanagari script was performed using syllabification approach that converted English into Hindi language. In the query labeling subtask, identification of English and Hindi words was performed using a hybrid approach that involved morphological analysis of English words and a corpus based approach to identify frequently occurring Hindi words. In the multi-script adhoc retrieval of Hindi song lyrics subtask, the queries were formulated that contained both Roman and Devanagari script and Roman script for separate run submissions. The evaluation of our experiments achieved a higher recall value of query labeling in subtask-1 however the results of subtask-2 are indicating average performance.

Keywords: Machine Transliteration, Query Labeling, Information retrieval, Syllabification.

1 Introduction

Now a day's there is need to provide local language support in web based applications because various domains such as ecommerce sites require English knowledge. India is a multilingual country with 22 constitutional officially recognized languages and 11 different scripts used in different regions spread across the country [1]. Hindi is the official national language of India and world's fourth most commonly used language after Chinese, English and Spanish, derived from the Sanskrit and use the "Devanagari" script for writing. Transliteration between Devanagari script and Roman script is difficult due to difference in writing script, difference in number of alphabets, capitalization of leading characters, phonetic characteristics, character length, various transliterations [2].

Transliteration tasks become difficult in presence of out of vocabulary words and noisy words which are present in the corpus of documents. Out of vocabulary (OOV) words are problematic in cross-lingual information retrieval. The OOV words are name entities, number, technical terms and acronyms. Transliteration is totally different from translation for instance, English word "world" when translated into Hindi language become "दुनिया". This is different from transliterated in which the word "Duniya" would map to "दुनिया". The challenge in transliteration is take the word "राष्ट्रपति" for this word "rashtrapati", "rashtrapathi", "raashtrapathy", "raashtrpati" are various possible combinations may possible which one should be correct is again an issue. This paper focus on the methods that we have applied in the shared task of transliterated search.

In this paper, section 2 focuses on literature review of transliteration from Roman script to Devanagari script, section 3 describes the syllabification approach used in our experiments, section 4 contains describes techniques applied in subtask-1 and its evaluation, section 5 contains the techniques applied in subtask-2 and its evaluation, finally section 6 concludes the results.

2 Related Work

Since past few years, there has been noticeable development in the area of machine transliteration especially for Roman script and Devanagari. Earlier approaches include adhoc bilingual task done for Hindi and Marathi to English language [3]. They used a query translation based approach using bi-lingual dictionaries. Query words which are absent in dictionary are transliterated using rule based approach which utilizes the corpus to return the 'k' closest English transliterations of given Hindi or Marathi word. The resulting multiple transliteration choices for each query word are disambiguated by using the page rank algorithm which is based on term-term co-occurrence statistics and produces the final translated query.

Rama T. and Gali K. addressed the transliteration problem as a translation problem [4]. They used SMT system, GIZA++, beam search based decoder for developing the transliteration model, they applied English-Hindi aligned word corpus to train and test the system. Transliteration system developed by Amitava Das et al. was based on news corpus as a part of Machine Transliteration Shared Task training datasets [5] after which the letter-phoneme technology to improve the performance of existing model. An informed Phrase-based statistical machine translation, considered the translating characters rather than words in character-level translation systems [5]. A memory-based classification framework enables estimation of features to avoid data sparseness problems [6]. The character, transliteration unit level and position are dependent source context properties which produces significant improvement in evaluation metrics. In this way, the problem of Machine transliteration was removed by adding source context modeling into state-of-the-art log-linear phrase-based statistical machine translation and increased the system performance. Malik A. et. Al proposed an Urdu to Hindi Transliteration using hybrid approach in 2009 [5]. They have choosen the hybrid approach of finite-state machine based techniques combined with the statistical word language model to improve the quality of results.

A Punjabi-Hindi transliteration system was developed based on statistical approach [11]. The system used letter to letter mapping as baseline and try to find out the improvements by statistical methods. They used a Punjabi-Hindi corpus for training and SMT tools in it. Online Hindi Lyrics the Hindi-English Transliteration research was done using noise removal and then by applying an EW-HMM model to align the Devanagari and Roman script songs with the word level. Then they applied the trained CRF engine to it.

A transliteration attempt was made for English-Hindi language pair consisting of Indian names using different character encoding for target language ,by using phrase based statistical machine translation technique [13]. Khapra proposed [14] a framework for transliteration which uses a word-origin detection engine, CRF based transliteration engine, re-ranking model based on lexicon lookup to improve the transliteration accuracy.

3 Syllabification Approach used for Machine Transliteration

In our approach we used backward transliteration that involved transliteration from Roman script to Devanagari script. Statistical machine learning approach was used for transliteration while TF-IDF model was used for Information retrieval. We used the syllable theory for transliteration [7]. The syllabification approach is as described below.

Linguists have different languages have constraints on possible consonant and vowel sequences that characterize not only the word structure for the language but also the syllable structure. In the syllabification approach, vowels are at center known as nucleus, consonants are at beginning, termed onset and at the end is coda. The syllable structure is shown in figure 1.

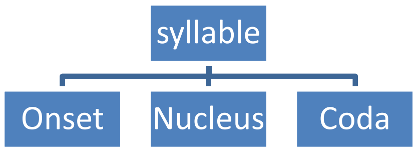

Figure1: Syllable Structure

The input language string is broken up into syllables according to rules specific to the source and target languages. An example showing syllabification of a word - “Word” is shown in figure 2.

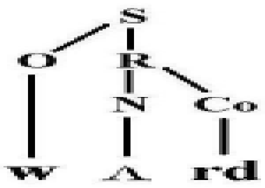

Figure2: “Word” mapped onto syllable structure

There are some rules for auto syllabification [15]. Such as, a, e, i, o, u are defined as vowels and y is defined as a vowel when it is not followed by any vowel. Rest characters taken as consonants. When m and n are surrounded by vowels then duplicate the nasals and if appear after a vowel, combine with that vowel to form a new vowel. Consecutive consonants are separated and consecutive vowels are treated as a single vowel. A consonant and a following vowel are treated as a syllable. Each isolated vowel or consonant is regarded as an individual syllable.

The training set used large parallel corpora of names written in both English and Hindi languages. applied syllabification to that names, either by a rule-based system or by a statistical system. For each syllable string of English, they store the number of times any Hindi syllable string is mapped to it. In terms of probability any Hindi syllable string is mapped to any English syllable string. Viterbi Algorithm was applied to find out six most probable transliterated words with their corresponding probabilities. However we have considered the top two transliterated words in subtask-1 and the topmost word in subtask-2.

The system was trained over character-aligned parallel corpus containing nearly 22,500 pairs. The character or pair of characters is transliterated as:

Source	**Target**
s h i v	श ◌ि व
m a d h a v	म ◌ ◌ा ध व
m o h a m m a d	म ◌ ◌ो ह म ◌ ◌् म द
j a y a n t e e d e v i	ज य ◌ ◌ं त ◌ ◌ी द ◌ ◌े व ◌ ◌ी

Having useful phrase transliterations and built a language model over the target side characters, these two components are given weights and combined during the decoding of the source name to the target name. Decoding builds up a transliteration from left to right and not allowing for any reordering the foreign characters to be transliterated are selected from left to right as well, computing the probability of the transliteration incrementally. Decoding start with no source language characters having been transliterated, called an empty hypothesis, and then expands this hypothesis, to make other hypotheses covering more characters. A source language phrase f_i to be transliterated into a target language phrase e_i is picked, this phrase must start with the left most character of their source language name that has yet to be covered, and potential transliteration phrases are looked up in the translation table. The evolving probability is computed as a combination of language model, looking at the current character and the previously transliterated n−1characters, depending on n-gram order, and transliteration model probabilities.

The hypothesis stores information of source language characters have been transliterated, the transliteration of the hypothesis expansion, the probability of the transliteration up to this point and a pointer to its parent hypothesis. All source languages get covered. The chosen hypothesis is the one which covers all foreign characters with the highest probability. The flow of syllabification approach is as show in figure 3.

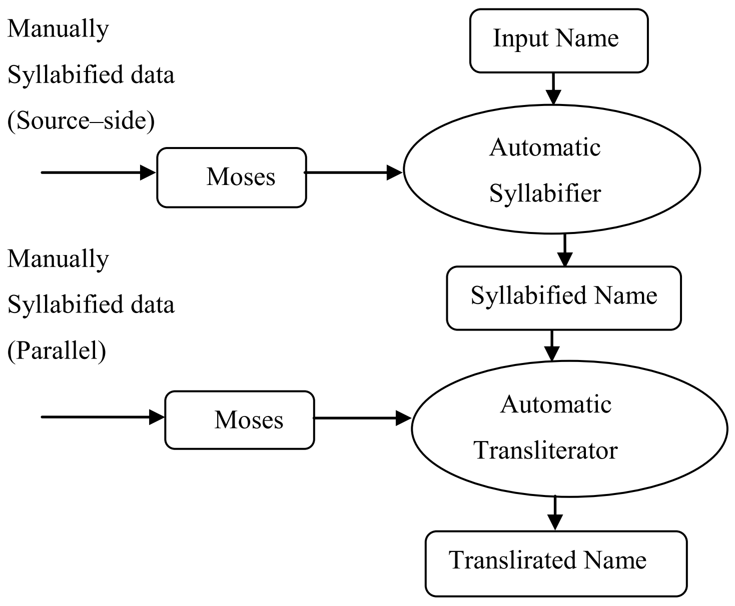

Figure3: Syllabification approach for Machine Transliteration

The final transliteration is constructed by backtracking through the parent nodes in the search that lay on the path of the chosen hypothesis. To search the space of possible hypotheses exhaustively is unfeasible and Moses [8] employs a number of techniques to reduce this search space. The Algorithm is as follows:

Step-1: Identify first nucleus in the word it may be single vowel or an occurrence of consecutive vowels.

Step-2: All the consonants before this nucleus will be parsed as the onset of the first syllable.

Step-3: Find next nucleus in the word. If do not find another nucleus then parse the consonants to right of current nucleus as the coda of the first syllable, else move to the next

Step-4: Now work on the consonant cluster there in between two nuclei. Divide these consonants in two parts, one serving as the coda of the first syllable and the other serving as the onset of the second syllable.

Step-5: If the no. of consonants in the cluster = 1, go to an onset of the second nucleus as per the Maximal Onset Principle and Constrains on Onset.

Step-6: If the no. of consonants in the cluster = 2, check whether both of these can go to the onset of the second syllable as per the allowable.

```
  If (two-consonant cluster = legitimate onset)
   {(Serve as the onset of the second syllable)}
  Else
   {(first consonant = coda of the first syllable,
      second consonant = onset of the second syllable)}
```

Step-7: If the no. of consonants in the cluster = 3, check whether all three will serve as the onset of the second syllable, if not, then check for the last two, if not, then parse only the last consonant as the onset of the second syllable.

Step-8: If the no. of consonants in the cluster > 3, except the last three consonants. Parse all the consonants as the coda of the first syllable as we know that the maximum number of consonants in an onset can only be 3. With the remaining 3 consonants, apply the same algorithm as in step 7.

Step-9: After having consonants among the coda of the previous syllable and the onset of the next syllable, we truncate the word till the onset of the second syllable and assuming this as the new word, we apply the same set of steps on it.

4 Query Labeling Subtask

In the query labeling subtask, we assumed that all terms are either in English or in Hindi language written in Roman script. We tried to identify the English terms and label them, rest of the terms were labeled as Hindi terms. The experiments went through two passes. In the first pass we identified the terms and labeled each term while in second pass the terms labeled with 'H' were transliterated into Devanagari script using the syllabification approach. The algorithmic steps followed in term labeling are as follows:

Step-1: Tokenize the terms from query set and iterate steps 2 to 4

Step-2: For each term perform spell-check for English language, if no spell error then label the word as English and consider next token else goto step-3

Step-3: For each term, perform morphological analysis and find a match with English dictionary, if the match is found then label the word as English and consider next token else goto step-4

Step-4: For each term, get the frequency count of that term, "Fc" from foreign corpus and frequency count "Lc" from local corpus; if Fc > Lc then label the word as English else label the word as Hindi

We used two different news corpus for step-4. Both the news corpuses were derived from the same year of publication and were normalized by disk size. The foreign news corpus included news collection of US and UK whereas the local news corpus collection was used from FIRE-2011 dataset. The rationale behind using two different news corpuses was to identify ambiguous terms like "Obama" that might be wrongly classified as Hindi term in our algorithmic approach.

We submitted two runs for the query labeling subtask. Query labeling remained same in both the runs; however the transliteration approaches were different for both the runs. In the first run, the terms derived from the transliteration model considered a higher probability when compared to the second run. The results of subtask-1 are shown in Table 1.

Language	**Metric**	**Run-1**	**Run-2**	**Maximum**	**Median**
Hindi	Exact query match fraction	0.0360	0.0020	0.1960	0.0290
10 runs	Exact transliteration pairs	815/1852	282/1850	N. A.	N. A.
5 teams	Transliteration-precision	0.4354	0.1508	0.7656	0.4121
#(True \H) =	Transliteration-recall	0.3336	0.1154	0.7646	0.3903
#(True \E) =	Transliteration-Fscore	0.3778	0.1308	0.7651	0.3867
#(\N) = 232	Labelling accuracy	0.8102	0.8096	0.9842	0.9540
N = Names	Eng-precision	0.5616	0.5607	0.9654	0.9302
and ambiguities	Eng-recall	0.9743	0.9743	0.9743	0.9627
excluded from	Eng-Fscore	0.7125	0.7118	0.9672	0.9019
Analysis	L-precision	0.9893	0.9893	0.9902	0.9878
	L-recall	0.7581	0.7573	0.9889	0.9791
	L-Fscore	0.8584	0.8579	0.9896	0.9700

Table 1: Results of Subtask-1

From the results, it can be inferred that we got a higher English labeling accuracy as the first pass of our experiments focused on morphological analysis of English language. However, the algorithm was not able to identify terms like “youtube”, “indian”, “jones” as they were absent in the dictionary and news corpus and were wrongly classified as Hindi terms.

5 Results And Error Analysis of subtask-2

We retrieved the Hindi song lyrics which are written in the Roman script or Devanagari script. We used the Hindi song lyrics corpus from the FIRE 2013. The dataset contained 62,888 Hindi song lyrics. Queries were provided by the FIRE 2013 as a part of transliterated search task. Out of total 50 queries, 25 queries were made available for the training purpose and remaining 25 were used for the test. We submitted two runs; the first run contained queries formulated using the Roman script and Devanagari script, while the second run contained queries in Devanagari script.

The transliteration of queries was done using syllabification approach [7]. For each term, the model provides six different transliteration terms in the decreasing order of their probabilities. In this work, we have considered the topmost transliteration term with highest probability. Later the Hindi song lyrics corpus was indexed and retrieval was performed using the test queries using TF-IDF model. In first run, we used the query which contains both the formats in it, that is Roman script and the Devanagari script in it. Example is “मेरे सापनोन कि रानी काब् आयेगी तु mere sapnon ki rani kab aayegi tu”. For second run, we used the query which is in the roman script only. Example is “mere sapnon ki rani kab aayegi tu”. As we have two results with us by comparing with the FIRE 2013 results we got the following measurements. Our Run1 results are better than the Run2. In the following result table the nDCG is normalized discounted cumulative gain. The MAP is mean average precision and the MRR is mean reciprocal rank. The results are shown in Table 2.

Metrics	**Run-1**	**Run-2**	**Maximum Score**	**Median Score**
nDCG@5	0.5627	0.5262	0.8052	0.5620
nDCG@10	0.5619	0.5232	0.8002	0.5608
MAP	0.2546	0.2163	0.4236	0.2355
MRR	0.5835	0.5730	0.8440	0.5884

Table 2: Results of subtask-2

From the results, we observed that still there are some problems in the transliteration which decreased the precision. Most common issues observed are error in the maatra such as for word “sapnon” we got “सापनोन”, “ki” is “की” & for “kab” it is “काब”. Then for “main” we got “मिन” and “mein” we got “मीन” but both the transliterations are wrong. For the word na => न and ka => क , the maatra was missed. Then multiple mapping of the words e.g. for t = त, ट, i.e. tera=>टेरा, tum => तूम, to => टो, teri =>टेरि . Also found the problem in missing sounds (फ, ख, छ ‘chh’, ksh) for them exact English word not available i. e. for word “accha” we got “आक्का”, for , “poochho” we got “पूछोट”. Multiple Transliterations- c,k problem is also there. The vowel are not giving perfect answers i.e. for “lo” we got “लॉ” , for ho we got “होर”, for “ko” we got “कॉ”. So, there is a need to work on the vowels. Allophones (wadala,vadala), Spelling Variations(shree,shri), Conjuncts formation, Incorrect Syllabification (gay tri / ga yat ri), for “kaise” is “कय्से” , “ikraar” is “इक्राचंद्र”, “kya” is “केया” are also some points we observed.

I. CONCLUSION

In this paper, we have presented our techniques applied to subtask-1 and subtask-2 respectively. The focus was on transliteration from Roman script to Devanagari script. We used the syllabification approach and considered the most probable term in the transliteration process. The word labeling task was performed assuming that a term either belongs to English language or Hindi language. We were able to get high accuracy in English recall as the labeling approach used morphological analysis and dictionary approach. However due to syllabification model, the transliteration did not give high precision resulting in lower precision of transliteration tasks and subsequently lower precision metrics in the song lyrics retrieval tasks.

REFERENCES

1. Jain M., Kumar V. and Kumar M.S., Encoding of Indic script : ISCII & Unicode, Department of Information Technology, pp. 95-106, New Delhi
2. Dhore M., Dixit S. and Dhore R., “Hindi and Marathi to English NE Transliteration” : LINGUISTIC RESEARCH GROUP, VIT, pp. 111-117, Pune, Maharashtra, India
3. Kumar M., Chinnakotla, Ranadive S., Bhattacharyya P., and Damani O., “Hindi and Marathi to English Cross Language Information Retrieval” at CLEF (2007)
4. Rama T., Gali K., ‘Modeling Machine Transliteration as a Phrase Based Statistical Machine Translation Problem’, Language Technologies Research Centre, IIIT, Hyderabad, India. (2009)
5. Das A., Ekbal A., Mandal T. and Bandyopadhyay S., “English to Hindi Machine Transliteration System at NEWS’, Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration. Association for Computational Linguistics, (2009)
6. Antony P J and Dr. Soman K P , ‘Machine Transliteration for Indian Languages: A Literature Survey’, International Journal of Scientific & Engineering Research , Volume 2, Issue 12, December-2011, (2011)
7. Aggarwal A., ‘Transliteration involving English and Hindi languages using Syllabification Approach’, M.Tech thesis, Indian Institute of Technology, Bombay, January (2010).
8. Moses Tool, http://www.statmt.org/moses/
9. GIZA++ Statistical machine translation toolkit, http://code.google.com/p/giza-pp/
10. The SRI language modelling toolkit, http://www.speech.sri.com/projects/srilm/
11. Singh G., Kaur J., “Punjabi to Hindi Statistical Machine Transliteration”, International Journal of Information Technology and Knowledge Management, Volume 4, No. 2, pp. 459-463 (2011)
12. Gupta K., Choudhury M., Bali K., “Mining Hindi-English Transliteration Pairs from Online Hindi Lyrics”, LREC (2012)
13. Sharma S., Bora N. and Halder M., “English-Hindi Transliteration using Statistical Machine Translation in different Notation”, International Conference on Computing and Control Engineering (2012)
14. Khapra M., Bhattacharyya P., “Improving transliteration accuracy using word-origin detection and lexicon lookup”, Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration. Association for Computational Linguistics, (2009)