ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

Transliteration from English to Indian Languages based on AnglaMT Machine Translation Perspective

Jayan V, Bhadran V K

Language Technology Section
Centre for Development of Advanced Computing Thiruvananthapuram
jayan@cdac.in

Abstract - **In this paper we discuss about the hybrid approach for the English-Malayalam Transliteration. A rule based approach for the Machine Translation (MT) system requires huge database in the form of dictionaries and rules. It is difficult to accommodate all the form of data in to the dictionaries. The product names, proper names, technical terms, etc. may not have a target language equivalent. Whenever we come through the Machine Translation applications there is always a possibility of occurrence of unknown words. The words that are not found in the dictionaries are treated as the unknown words. All these terms have to be transliterated. It is difficult task to inculcate all the pronunciations for the unknown words. There are many factors that affect the transliteration.**

Keywords—Named Entity, Grapheme, Disambiguation

I. Introduction

Named entity (NE) transliteration is the process of transcribing a NE from a source language to some target language while preserving its pronunciation in the original language. Automatic NE transliteration is an important component in many cross-language applications, such as Cross-Lingual Information Retrieval (CLIR) and Machine Translation (MT) (Hermjakob et al., 2008; Klementiev and Roth, 2006a; Meng et al., 2001; Knight and Graehl, 1998)[5]. Transliteration is an easy task if it is concerned with the same language family to some extent, which requires only a phonetic mapping between character sets. However mere mapping of every source language grapheme to its target language counterpart may not work well. In practice this mapping depends on the context the characters appear in and on transliteration conventions used, which may change across domains. As a result, current approaches employ machine learning methods by giving enough labeled training data to learn how to determine whether a pair of words constitutes a transliteration pair. These methods typically require training data and language-specific expertise which may not exist for many languages. In this paper we try to overcome these difficulties and show that when the problem is modeled correctly, a simple character level mapping is a sufficient resource. In our experiments, English was used as the source language, allowing us to use Romanization tables, a resource commonly-available for many languages. These tables contain an incomplete mapping between character sets, mapping every character to its most common counterpart.

When we developed a Transliterator for the MT system using certain transliteration schemes that are generally followed for English-Malayalam transliteration we were able to achieve only an accuracy of about 50%. Later we added rules which are developed by context based transliteration method. This approach facilitated us an accuracy of about 74%, an improvement of about 24%. Usually English is not read as it is written. This creates a great difficulty in transliteration. The importance of transliteration is prominent when we look into a perfectly translated sentence becoming unreadable due to improper transliteration. We followed a method that used a Grapheme mapping of source language and target language and then used exceptions as rules. Some cases cannot be handled even by using exceptions. For that we have to use the database. The names like 'Sasi' cannot be transliterated as we needed. The intended translation is 'SaSi'(ʃaʃi) in Malayalam.

The changes occurring in the pronunciation depends on the following factors

- Position of a character in the word
- Type of adjacent characters
- Presence of one particular character at any position
- Socio-cultural background of a speaker
- Origin of the word

Four machine transliteration models have been proposed by several researchers: (a) grapheme- based transliteration model (GT) (Lee & Choi, 1998; Jeong, Myaeng, Lee, & Choi, 1999; Kim, Lee, & Choi, 1999; Lee, 1999; Kang & Choi, 2000; Kang & Kim, 2000; Kang, 2001; Goto, Kato, Uratani, & Ehara, 2003; Li, Zhang, & Su, 2004)[2], (b) phoneme - based transliteration model (PT) (Knight &Graehl, 1997; Lee, 1999;

Jung, Hong, & Paek, 2000; Meng, Lo, Chen, & Tang, 2001)[3], (c) hybrid transliteration model (HT) (Lee, 1999; Al - Onaizan & Knight, 2002; Bilac & Tanaka, 2004), and (d) correspondence-based transliteration model (CT) (Oh & Choi, 2002). These models are classified with respect to the units to be transliterated. The GT is normally referred to as the direct method because it directly transforms source language graphemes into target language graphemes without any phonetic knowledge of the source language words. The PT usually needs two steps: 1) produce source language phonemes from source language graphemes; 2) produce target language graphemes from source phonemes. The HT and CT make use of both source language graphemes and source language phonemes when producing target language transliterations. Hereafter, we refer to a source language grapheme as a source grapheme, a source language phoneme as a source phoneme, and a target language grapheme as a target grapheme. The transliterations produced by the four models usually differ because the models use different information. Generally, transliteration is a phonetic process, as in PT, rather than an orthographic one, as in GT (Knight & Graehl, 1997). However, standard transliterations are not restricted to phoneme-based transliterations. For example, the standard Malayalam transliterations of “data, amylase, and neomycin” are, respectively, the phoneme based transliteration ‘dE-ta’, the grapheme-based transliteration ‘a-mi-le:s’, and ‘ni-yo:-mai-sin’, which is a combination of the phoneme-based transliteration ‘ne-o’ and the grapheme-based transliteration ‘mai-sin_’. Furthermore, if the unit to be transliterated is restricted to either a source grapheme or a source phoneme, it is hard to produce the correct transliteration in many cases. For example, PT cannot easily produce the grapheme-based transliteration ‘a-mi-lEs, the standard Malayalam transliteration of amylase, because PT tends to produce ‘a-mai-lEs’ based on the sequence of source phonemes /A MAI LES/. Multiple transliteration models should therefore be applied to better cover the various transliteration processes. To date, however, there has been little published research regarding a framework in which multiple transliteration models can operate simultaneously. Furthermore, there has been no reported comparison of the transliteration models within the same framework and using the same data although many English-to-Korean transliteration methods based on GT have been compared to each other with the same data (Kang & Choi, 2000; Kang & Kim, 2000; Oh & Choi, 2002).

II. Related Works

Machine transliteration has received significant research attention in recent years especially in the cross lingual information extraction field. In most cases, the source language and target language have been English and an Asian language, respectively – for example, English to Japanese (Goto et al., 2003), English to Chinese (Meng et al., 2001; Li et al., 2004), and English to Korean (Lee & Choi, 1998; Kim et al., 1999; Jeong et al., 1999; Lee, 1999; Jung et al., 2000; Kang & Choi, 2000; Kang & Kim, 2000; Kang, 2001; Oh & Choi, 2002). Many transliteration developments are also happened in India and are available online. But there is no significant improvement in the accuracy and sometimes not at all give acceptable transliteration. In this section, we review previous work related to the four transliteration models.

A. *Grapheme-based Transliteration Model*

Conceptually, the GT is direct orthographical mapping from source graphemes to target graphemes. Several transliteration methods based on this model have been proposed, such as those based on a source-channel model (Lee & Choi, 1998; Lee, 1999; Jeong et al., 1999; Kim et al., 1999), a decision tree (Kang & Choi, 2000; Kang, 2001), a transliteration network (Kang & Kim, 2000; Goto et al., 2003), and a joint source-channel model (Li et al.,2004). The source-channel model deals with the English-Korean transliteration. They use a chunk of graphemes that can correspond to a source phoneme. The transliteration processes are described below:

- English words are segmented into a chunk of English graphemes.
- All possible chunks of Korean graphemes corresponding to the chunk of English graphemes are produced.
- The most relevant sequence of Korean graphemes is identified by using the source-channel model.

The advantage of this approach is that it considers a chunk of graphemes representing a phonetic property of the source language word. However, errors in the first step (segmenting the English words) propagate to the subsequent steps, making it difficult to produce correct transliterations in those steps. Moreover, there is high time complexity because all possible chunks of graphemes are generated in both languages. In the method based on a decision tree, decision trees that transform each source grapheme into target graphemes are learned and then directly applied to machine transliteration. The advantage of this approach is that it considers a wide range of contextual information, say, the left three and right three contexts. However, it does not consider any phonetic aspects of transliteration. Kang and Kim (2000) and Goto et al. (2003) proposed methods based on a transliteration network for, respectively, English-to-Korean and English-to-Japanese

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

transliteration. Their frameworks for constructing a transliteration network are similar – both are composed of nodes and arcs. A node represents a chunk of source graphemes and its corresponding target graphemes. An arc represents a possible link between nodes and has a weight showing its strength. Like the methods based on the source-channel model, their methods consider the phonetic aspect in the form of chunks of graphemes. Furthermore, they segment a chunk of graphemes and identify the most relevant sequence of target graphemes in one step. This means that errors are not propagated from one step to the next, as in the methods based on the source-channel model. The method based on the joint source-channel model simultaneously considers the source language and target language contexts (bigram and trigram) for machine transliteration. Its main advantage is the use of bilingual contexts.

B. Phoneme-based Transliteration Model

In the PT, the transliteration key is pronunciation or the source phoneme rather than spelling or the source grapheme. This model is basically source grapheme-to-source phoneme transformation and source phoneme-to-target grapheme transformation. Knight and Graehl (1997) modelled Japanese-to-English transliteration with weighted finite state transducers (WFSTs) by combining several parameters including roman-to-phoneme, phoneme-to-English, English word probabilities, and so on. A similar model was developed for Arabic-to-English transliteration (Stalls & Knight, 1998). Meng et al. (2001) proposed an English-to-Chinese transliteration method based on English grapheme-to-phoneme conversion, cross-lingual phonological rules, mapping rules between English phonemes and Chinese phonemes, and Chinese syllable-based and character-based language models. Jung et al. (2000) modelled English-to-Korean transliteration with an extended Markov window. The method transforms an English word into English pronunciation by using a pronunciation dictionary. Then it segments the English phonemes into chunks of English phonemes; each chunk corresponds to a Korean grapheme as defined by handcrafted rules. Finally, it automatically transforms each chunk of English phonemes into Korean graphemes by using an extended Markov window. Lee (1999) modeled English-to-Korean transliteration in two steps. The English grapheme-to-English phoneme transformation is modeled in a manner similar to his method based on the source-channel model described in Section 2.1. The English phonemes are then transformed into Korean graphemes by using English-to-Korean standard conversion rules (EKSCR) (Korea Ministry of Culture & Tourism, 1995). These rules are in the form of context-sensitive rewrite rules, “PaPxPb → y”, where y represents Korean grapheme , meaning that English phoneme Px is rewritten as Korean grapheme y in the context Pa and Pb, where Px, Pa, and Pb represent English phonemes. For example, In the case of English Malayalam Transliteration, “Pa = *, Px = /y/, Pb = end → ‘i’” means “English phoneme /y/ is rewritten into Malayalam grapheme ‘i’ if it occurs at the end of the word (end) after any phoneme (*)”. This approach suffers from both the propagation of errors and the limitations of EKSCR. The first step, grapheme-to-phoneme transformation, usually results in errors, and the errors propagate to the next step. Propagated errors make it difficult for a transliteration system to work correctly. In addition, EKSCR does not contain enough rules to generate correct Korean transliterations since its main focus is mapping from an English phoneme to Korean graphemes without taking into account the contexts of the English grapheme.

C. Hybrid and Correspondence-based Transliteration Models

In order to accommodate the use of both source graphemes and source phonemes in machine transliteration led to the correspondence-based transliteration model (CT) (Oh & Choi, 2002) and the hybrid transliteration model (HT) (Lee, 1999; Al-Onaizan& Knight, 2002; Bilac& Tanaka, 2004). The former makes use of the relationship between a source grapheme and a source phoneme when it produces target language graphemes; the latter simply combines GT and PT through linear interpolation. Note that the HT combines the grapheme-based transliteration probability (Pr(GT)) and the phoneme-based transliteration probability (Pr(PT)) using linear interpolation. Oh and Choi (2002) taken up the contexts of a source grapheme and its corresponding source phoneme for English-to-Korean transliteration. They used EKSCR as the basic rules for their method. Additional contextual rules are also semi-automatically constructed by examining the cases in which EKSCR produced incorrect transliterations. It is because of a lack of contexts. These contextual rules are in the form of context-sensitive rewrite rules, “CaCxCb → y”, meaning “Cx is rewritten as target grapheme y in the context Ca and Cb”. Note that Cx, Ca, and Cb represent the correspondence between the English grapheme and phoneme. For example, in Malayalam, we can read “Ca = (* : /Vowel/),Cx = (r : /R/),Cb = (* : /Consonant/) → NULL” as “English grapheme r

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

corresponding to phoneme /R/ is rewritten into null Malayalam graphemes when it occurs after vowel phonemes, (∗ : /V owel/), before consonant phonemes, (∗ : /Consonant/)". The main advantage of this approach is the application of a sophisticated rule that reflects the context of the source grapheme and source phoneme by considering their correspondence. However, there is lack of portability to other languages because the rules are restricted to Malayalam, but there can be possibility of applying rules to same language family members. Several researchers (Lee, 1999; Al-Onaizan& Knight, 2002; Bilac& Tanaka, 2004) have proposed hybrid model-based transliteration methods. They model GT and PT with WFSTs or a source-channel model and combine GT and PT through linear interpolation. In their PT, several parameters are considered, such as the source grapheme-to-source phoneme probability, source phoneme-to-target grapheme probability, and target language word probability. In their GT, the source grapheme-to-target grapheme probability is mainly considered. The main disadvantage of the hybrid model is that the dependence between the source grapheme and source phoneme is not taken into consideration in the combining process; in contrast, Oh and Choi's approach (Oh & Choi, 2002)[2] considers this dependence by using the correspondence between the source grapheme and phoneme.

III. TRANSLITERATION RULES

The transliteration scheme lists the most possible literal pronunciation of characters and thus they will not be a suitable one in all cases. When a particular character occur in combination with other character or depending on its position in a word the change in pronunciation occur. The words with all the characters in capital are get transliterated character by character as given in example below:

IIT →aį aį ti

CDAC →si di e si

Consider the first alphabet 'a' in English, We deduced the following rules.

Rule 1:

If 'a' followed by another 'a' then the transliteration will be a long 'A', for eg:

Aadhar → Adhar(ആധർ)

Rule 2:

If 'a' followed by 'e' and is the pre-final character, then the transliteration will be 'e'

Vitae →viṛṛe(വിറ്റെ)

Rule 3:

If 'a' is followed by 'e' and is occurring at the initial position of the word, then the transliteration will be 'ey'.

Aeroplane→eyṛOpleyṇ (എയ്റോപ്ലെയ്ൻ)

Rule 4:

If 'a' is followed by 'i' at the word end then the transliteration will be 'Ayi' as in

Rai→ṛAyi(റായി)

Rule 5:

If 'a' is followed by 'i' in other cases the transliteration will be 'Ay' as in

Taiwan →tAyvAṇ (തായ് വാൻ)

Rule 6:

If 'a' is followed by 'u' in the word then the transliteration will be 'O' as in

Australia →OstrEliya(ഓസ്ട്രേലിയ)

Rule 7:

There are 3 condition for the 'ao' combination. If 'a' is followed by 'o' and is placed in the pre-final position of the word then it is transliterated as 'Avu' as in

Rao→ṛAvu(റാവു)

In another condition if 'ao' occurs in the initial position of the word then the transliteration will be 'ayO' as in

Aorta →ayOrtta(അയോർട്ട)

In all other condition the transliteration of 'ao' combination is 'AvO' as in

Taos →tAvOs(ടാവോസ്)

Rule 8:

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

There are three conditions for the 'ay' combination. In first case 'ay' in general transliterated as 'Ey' as in

Raygan→ŗEygan(റേയ്ഗൻ)

In the second case if the combination comes at the word end then the transliteration will be 'ay' only as in

Vijay →vijay(വിജയ്)

In the third case if the combination is followed by any vowels then the transliteration will be 'ay' as in

Amaya →amaya(അമയ)

Similarly we generated rule for different combination of English alphabets starting from A to Z. This methodology has improved the accuracy of the transliterator gradually. Still the system showed some drawbacks in transliteration. This is also not providing a complete solution. There are some cases that produce error transliteration by using this method.

Now we discuss on the context dependent transformation of the grapheme to phoneme. This can be done with the help of context based model. There are some instances where the transliteration of particular character will also depend on the presence of some other character that is not adjacent to the character that we are considering. For example consider the two proper nouns 'Naveen' and 'Praveen',

Naveen →navIņ(നവീൻ)

Praveen →pravIŅ(പ്രവീൺ)

Both the words are having the same chunk ending, 'een'. Their transliteration will be 'ņ (ൻ)' and 'Ņ(ൺ)' respectively. Both are the 'chillu'1 characters. First is the dental-nasal stop consonant and second is the retroflex - nasal stop consonant. While retrospection through the words we found that the words with the existence of 'r' combined with a consonant at any position will affect the dental-nasal form and that will pronounced in retroflex-nasal form. Similarly in some other occasions the presence 'r' will result 't' in dental unaspirated voiceless plosive. In normal case it is transliterated as retroflex unaspirated voiceless plosive. Transliteration of this type of words should be done prior to the mapping with transliteration schemes already developed for a particular language.

There is another case that we have take in to consideration is the occurrence chunks. Consider the word 'Colgate', brand name of toothpaste; here the chunk 'gate' has many occurrences in some other occasions like tollgate, tailgate, etc. We should identify such 'chunks' and create a table for it. In every case the transliteration of the chunk 'gate' is same. So we found that such replacement works well in transliteration.

There are other instances where some consonant clusters fail to give proper transliteration. Consider the word 'Kunhamma' and 'inhibase' and 'allenhist', the transliteration is given below:

a) Kunhamma→kuňňamma(കുഞ്ഞമ്മ)

b) Inhibase→iņhibEs(ഇൻഹിബേസ്)

c) Allenhist→alaņhisŗŗ(അലൻഹിസ്റ്റ്)

In the case (b) and (c) 'n' and 'h' are transliterated separately but in the first case the transliteration of 'nh' taken in combination and got a separate transliteration. We have to deduce the transliteration for such terms by stochastic method.

This approach of transliteration can be followed in general for all the Indian languages. There are some exceptions for the languages like Bengali and Hindi. Schwa deletion is prominent in Hindi. So care must be taken to handle such cases by introducing separate modules. In Bengali Language most of the situations the character 'a' is pronounced as 'o'. Keeping the discussed approaches in general and can add new modules to handle the exceptional cases for other languages.

We followed three methodologies for the transliteration, namely, Probability sequencing, pronunciation estimation (Phoneme based Transliteration) and grapheme mapping. Initially the system will look for any stored pattern in the dictionary. If the system doesn't find any matching word then it will go for further mapping. There are three methodologies adopted here.

A. *Grapheme Based Transliteration*

For English Malayalam transliteration we developed a transliteration scheme based on the maximum probable mapping of source language grapheme and target language

1 In Malayalam, the Virama (Chandrakala) serves two purpose, as an Inherent Vowel Killer, and to denote half-u (Samvrutokaram). Hence, to avoid confusion, certain consonants have special forms called "Chillu" do denote the pure consonant. Modern Malayalam uses 5 chillu forms for ന, ണ, ല, ള, ര. historically, other consonants also had chilluforms, but they are not in current usage.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

grapheme. This approach will work well for the same language families. But for a language like English this will give wrong transliteration, because English is not read like the way it is written. This will affect the application that is using the transliteration system. In the case of Machine Translation, even if the system gives good translation for a particular sentence, the bad transliteration will affect the total quality of the system.

Based on this approach we had transliterated about 4000 proper names and we were able to achieve only 50% accuracy. This approach has a limitation in achieving the higher accuracy especially for English-Malayalam Transliteration system.

There are cases where the independent occurrence of a particular grapheme has entirely different grapheme representation in target language in different context.

TABLE 1: ENGLISH-MALAYALAM GRAPHEME MAPPING

English	Phonetic Notation (Malayalam)
a	ʌ (അ)
aa	ɑː(ആ)
i,ui	I (ഇ)
ee,ii,ei,ie,ea	iː(ഈ)
u	ʊ (ഉ)
oo,uu	uː(ഊ)
e,ae	e (എ)

Such cases cannot be handled entirely by grapheme mapping. We have to think of some other methodologies to resolve such occurrences. Consider the table 2 given below:

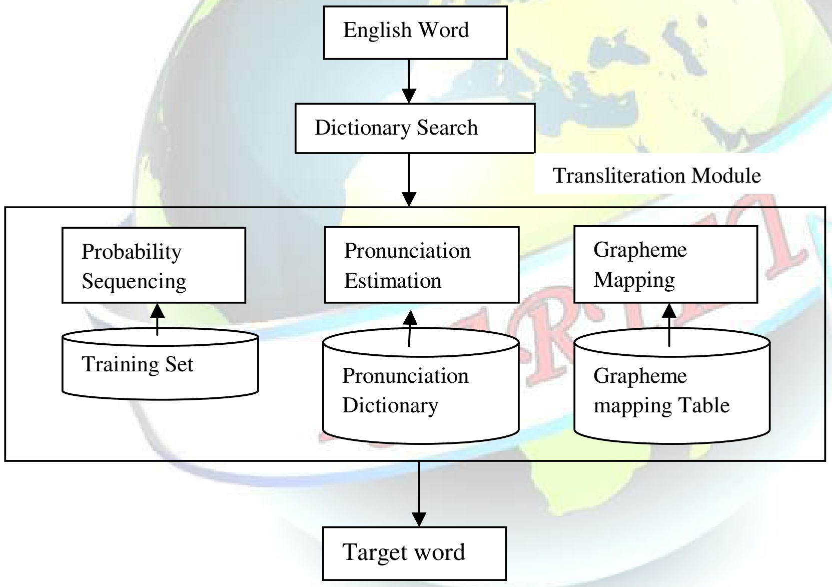

Fig 1: Schematic diagram of speech production.

TABLE 2: ENGLISH-MALAYALAM GRAPHEME MAPPING

Grapheme	Source Language Grapheme	Target Language Grapheme	Phonetic Notation
X (എക്സ്)	Dixit Xavier hexagon	ദീക്ഷിത് സേവിയർ ഹെക്സഗൺ	ǀ ðiːkʃɪθ ǀ ǀ sævIjar ǀ ǀ heksagʌn ǀ

The grapheme mapping for 'x' in source language and in target language is difficult because there are three possible grapheme representations for 'x' in target language. This is resolved by applying rules. When 'x' comes in the word start position, then it will pronounced as 's'. Other two cases are context dependent. That can be handled by other methodology. This led to the development of context dependent transliteration rules as discussed in section 3.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

B. Pronunciation Estimation

Normally we will have a dictionary of known words and their transliterated form will be stored in a dictionary. Whenever a new word comes the system goes for a pronunciation estimation module. Consider a source word Sw={s1,s2,…sn} and its pronunciation be Pw={p1,p2,…pk}, where n≥ k.

TABLE 3: PRONUNCIATION ESTIMATION TABLE

&	s1	s2	s3	s4	s5	#
M	i	c	h	e	a	l

There are some occasion where two vowels followed by 'ch', then the pronunciation will be 'kk', Similarly, we can apply some pronunciation estimation rules based on the adjacent graphemes. Here '&' represents start of the word and '#' represents end of a word. In this case if 'l' come at the end, it either pronounced as stop consonants of sonorant unaspirated voiced plosive 'l' and miscellaneous unaspirated voiceless plosive 'l'.

In table 4, the Source word contains 8 graphemes and its pronunciation contains 5 graphemes. Transliteration will be easier if we properly estimate the pronunciation. In the second name 'Micheal', 'ch' has different mapping. Hence the pronunciation also depends on context.

TABLE 4: ENGLISH-MALAYALAM GRAPHEME MAPPING

Source Input	Target output	Phonetic Representation
Michelle	മിഷേൽ	l miʃæḷ l
Micheal	മൈക്കൽ	l maıkkaḷ l
Chellamma	ചെല്ലമ്മ	l cellamma l
Lloyd	ലോയിഡ്	l lɒjid l

When 'll' comes in the middle gemination occurs. But in the first case when it appeared in the end followed by 'e', it pronounced as 'ḷ', which is a stop consonant. Similarly when it comes in the start it is pronounced as 'l', there is no gemination at the start of a word. Even this will not be a sufficient approach in some cases. A further more improvement can be achieved by adopting the probability sequencing. [6] discussed about a system,the effective incentive scheme is proposed to stimulate the forwarding cooperation of nodes in VANETs. In a coalitional game model, every relevant node cooperates in forwarding messages as required by the routing protocol. This scheme is extended with constrained storage space. A lightweight approach is also proposed to stimulate the cooperation.

C. Probability Sequencing

In some named entities there may be more than one possible combination of phoneme mapping. Disambiguation of this can be done using the probability sequencing.

Consider the table 5 showing probable transliteration of the word 'Lata'.

TABLE 5: PROBABILITY SEQUENCING

Sl No.	Transliterations	Phonetic Notations	Probability
1	ലത	l laða l	0.3
2	ലാത	l la:ða l	0.12
3	ലട	l lata l	0.2
4	ലാട	l la:ta l	0.05

In English, the vowel duration will depend on the class of graphemes following or preceding. Getting the phoneme for that particular grapheme will be difficult. So it would be better to get a probability sequencing of graphemes based on the stored patterns. In the table 5, we got some statistics for the proper name 'Lata'. Here the actual pronunciation will be the first one. It also got the maximum probability. There are some cases such that this combination will get some other phoneme representation as in the fourth one, la:ta. In addition to the example given above one more pronunciation is possible for't' as in the word 'later'. After getting the phoneme of maximum probability, it is mapped with the target language phoneme set. This will reduce the ambiguity.

It will work well for the domain specific transliteration. The medical terminologies will have some repeated graphemes as in achromycin, adriamycin, aureomycin, etc. The occurrence of the grapheme 'mycin' is prominent in medical domain. We can directly map the 'phoneme', 'maisin' to the word. This way the ambiguity in transliteration can be reduced.

IV. RESULTS

We had manually and automatically evaluated the system output and got an accuracy of 74% by using the hybrid approach of transliteration. There is an improvement of more than 20% as compared to grapheme based transliteration model. We developed this system as part of the English Malayalam Machine Aided Translation system. We tested the

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 6, April 2017

system for different domains like medicine, proper names, place names, etc. For manual evaluation we ranked the system based on the acceptability factor of a human evaluator. The acceptability score is given based on the scoring parameter given below:

No match at all

Differences in expected consonants and vowels and is difficult to understand

Major differences in long vowels and acceptable with difficult

Only minor differences in terms long and short vowels and is acceptable

Perfectly correct

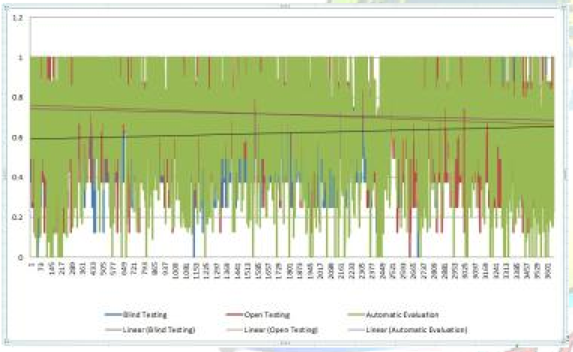

Fig 2: Transliteration accuracy plot.

We had considered here a set of 3600 proper nouns containing place names and human names. Here we had evaluated the transliterated output in three levels, blind testing, open testing and automatic evaluation. In blind testing the evaluator is provided with the transliterated output only. In open testing the evaluator is provided with the source and the transliterated output. In automatic evaluation the system generated a score by comparing the Actual transliteration and system generated transliteration. Here the automatic and the human evaluation had given 71.39% and 71.42% accuracy respectively. Further improvements can be achieved by doing deep research on the graphemes and their influence on the preceding and following graphemes. Hybrid approach improved the system accuracy from 51% to 72%. This improvement in accuracy also improved the readability of the translated output in MT system and thus accuracy of the MT system. The accuracy rate for different domain specific proper nouns is given below:

TABLE 6: ACCURACY OF TRANSLITERATION

Sl No	Domain	Accuracy
1	Medicine Name	81%
2	Place Name	65.69%
3	Human Name	75.89%
Overall Accuracy		**74.19%**

V. CONCLUSION

We had considered 6500 medicine names for calculating the transliteration accuracy. There is a significant drop in the accuracy of place names because the places names we used are contain some foreign names that is having an entirely different phoneme pattern. Domain customization is possible with this transliteration system because the MT system developed by us is domain specific, i.e. health domain. Here it gives the highest accuracy. Statistical Transliteration system will give much more accuracy with the help of huge samples.

VI. REFERENCES

[1]. Lee, J. S. and K. S. Choi, English to Korean Statistical transliteration for information retrieval.Computer Processing of Oriental Languages, 12(1), (1998), 17-37.
[2]. Jong-Hoon Oh and Key-Sun Choi,An Ensemble of Grapheme and Phonemefor Machine Transliteration, IJCNLP 2005, LNAI 3651, pp. 450 – 461, 2005.
[3]. Jong-Hoon Oh, Key-Sun Choi, Hitoshi Isahara, A Comparison of Different Machine Transliteration Models, Journal of Artificial Intelligence Research 27 (2006) 119–151
[4]. Manoj Kumar Chinnakotla and Om P. Damani, Character Sequence Modeling for Transliteration, Proceedings of ICON 2009: 7th International Conference on Natural Language Processing Macmillan Publishers, India.
[5]. Knight, K. and J. Graehl, "Machine Transliteration". In Proceedings. of the 35th Annual Meetings of the Association for Computational Linguistics (ACL), (1997)
[6]. Christo Ananth, M.Muthamil Jothi, A.Nancy, V.Manjula, R.Muthu Veni, S.Kavya, "Efficient message forwarding in MANETs", International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE), Volume 1,Issue 1, August 2015,pp:6-9
[7]. Kang B.J. and K-S. Choi, "Automatic Transliteration and Back-transliteration by Decision Tree Learning", In Proceedings of the 2nd International Conference on Language Resources and Evaluation, (2000)
[8]. Kang, I.H. and G.C. Kim, "English-to-Korean Transliteration using Multiple Unbounded Overlapping Phoneme Chunks", In Proceedings of the 18th International Conference onComputational Linguistics, (2000).