Computational Linguistics and Chinese Language Processing
Vol. 15, No. 2, June 2010, pp. 77-102

A Punjabi to Hindi Machine Transliteration System

Gurpreet Singh Josan*, and Gurpreet Singh Lehal*

Abstract

Transliteration is the general choice for handling named entities and out of vocabulary words in any MT application, particularly in machine translation. Transliteration (or forward transliteration) is the process of mapping source language phonemes or graphemes into target language approximations; the reverse process is called back transliteration. This paper presents a novel approach to improve Punjabi to Hindi transliteration by combining a basic character to character mapping approach with rule based and Soundex based enhancements. Experimental results show that our approach effectively improves the word accuracy rate and average Levenshtein distance of the various categories by a large margin.

Keywords: Transliteration, Punjabi, Hindi, Soundex Approach, Rule based Approach, Word Accuracy Rate.

1. Introduction

Every machine translation system has to deal with out-of-vocabulary words, like technical terms and proper names of person, places, objects, *etc*. *Machine transliteration* is an obvious choice for such words. When words cannot be found in translation resources, such as a bilingual dictionary, transliteration - the process of converting characters in one alphabet into another alphabet - is used. Transliteration is a process wherein an input string in some alphabet is converted to a string in another alphabet, usually based on the phonetics of the original word. If the target language contains all the phonemes used in the source language, the transliteration is straightforward, *e.g*. the Hindi transliteration of Punjabi word "ਕਮਰਾ" [kamarā] (room)[1] is "कमरा" [kamarā], which is essentially pronounced in the same way. Nevertheless, if some of the sounds are missing or are extra in the target language, they are generally mapped to the most phonetically similar letter, *e.g.*, in Hindi we have syllables (consonant clusters) that are written as a single atomic grapheme and have a double sound

* Department of Computer Science, Punjabi University Patiala, Punjab, India
Email: josangurpreet@rediffmail.com; gslehal@gmail.com
The author for correspondence is Gurpreet Singh Josan.

[1] The text in [ ] denotes phonetic transcription whereas in ( ) denotes the English translation of a word.

associated with them, like "क्ष", which is a combination of the sound of "क" and "ष". No such consonant cluster is present in Punjabi. So, a similar sounding letter generally is used to denote such sounds. Again, there is no rule to find out when a sequence of letters in Punjabi is going to map in a consonant cluster in Hindi, *e.g.,* consider the two names written in Punjabi, *viz*. ਸ਼ਿਤਿਜ [ṣitij] क्षितिज [kṣitija] and ਸ਼ਿਕਾਕਾਈ [śikākāī] शिकाकाई [śikākāī]. In the first name, the consonant cluster ਸ਼ਿ is mapped to क्षि, whereas, in the second name, the same is mapped to शि.

The important consideration in transliteration is to preserve the phonetics of transliterated word (Virga & Khudanpur, 2003). The transliteration process depends on the target language and on the education and experience of the actual transliterator. Thus, a single foreign word can have many different transliterations. For example, "ਮਹਿਫ਼ੂਜ਼" [mahiphūz] (safe) can be transliterated as "महफ़ूज़" [mahaphUz], "महफ़ूज़" [mahafUz], "महिफ़ूज़" [mahiphUz], "महिफ़ूज़" [mahifUz], *etc.* This variation, due to localization, poses problems in various NLP tasks, like cross language information retrieval. According to Viswanadha R. (2002), for a finer transliteration among two scripts, the scripts must have the features of completeness, predictability, pronounceable, unambiguousness, and partial reversibility. Transliteration schemes have to face the problem of letters present in one language and not in the other. Unless a superset of letters from all of the Indian Languages is formed, uniform transliteration is ruled out. Viswanadha, (2002) had the view that, when characters do not have any appropriate transliteration, they should be consumed and not replaced with any other character. This results in partial loss of reversibility and readability.

For several decades now, Roman transliteration has been used in English Indian language texts. Extensive research has been carried out on methodologies for transliterating Indian scripts to and from their Romanized counterpart. Transliteration among Indian scripts is a rather neglected area. With the development of ISCII (Indian Script Code for Information Interchange), the problems in transliteration among Indian scripts have been solved to some extent. ISCII has been designed via the phonetic property of Indian scripts and caters to the superset of all Indian scripts. The ISCII document is IS13194:1991, available from the office of the Bureau of Indian Standards. The ISCII approach is based on the fact that many Indian language groups share a common set of phonemes, and via the same set of codes for different Indian languages, it would be possible to change fonts from one language to the other, thereby "transliterating" the script in the process. By attaching an appropriate script rendering mechanism to ISCII, transliteration from one Indian script to another is achieved in a natural way. Later in the decade of the 1990s, Unicode standards were developed for representing the character set of all languages. In Unicode, alphabets are coded by script rather than language, which saves reduplication of the same character if it occurs in multiple languages. The Gurmukhi and Devanagari Unicode tables were based on templates from the 1988 versions of

the ISCII standard.

In this paper, we will discuss the Punjabi to Hindi Machine transliteration system. Although Punjabi and Hindi are related languages, and, except for a few cases, all letters of Gurmukhi script (a script for the Punjabi Language) are present in Devanagri script (a script for the Hindi Language), the task of transliteration from Punjabi to Hindi is not trivial. We will use letter to letter mapping as the **baseline** and try to find the improvements by rule based and Soundex based approaches. This work forms part of larger Punjabi to Hindi machine translation system, and the results of transliteration will be applied to this system for its improvement.

The remainder of the paper is organized as follows. Section 2 reviews previous work. In Section 3, the Gurmukhi and Devanagri scripts are compared, and the problems of transliteration from Punjabi to Hindi are discussed. Section 4 describes the formulation of the rule based model and the Soundex based model. Section 5 outlines the experiment and the evaluation methodology used. Section 6 will present the results. Finally, Section 7 contains our overall impressions and conclusions and points to future work.

2. Previous Work

The topic of machine transliteration has been studied extensively for several different language pairs, and many techniques have been proposed. Grapheme based models and Phoneme Based models are the two approaches found in literature. Grapheme refers to the basic unit of a written language: for example English has 26 graphemes or letters. Phonemes are the simplest significant unit of sound *e.g.* the /M/, /AE/, and /TH/ in *math.* Grapheme based models are classified into the statistical transliteration based model, decision tree based model, transliterated network based model *etc*. Some examples of this would be the noisy-channel model (NCM) (Lee & Chang, 2003; Virga *et al*., 2003), HMM (Jung, Hong & Paek, 2000), decision tree (Kang & Choi, 2000), transformation-based learning (Meng, Lo, Chen & Tan, 2001), statistical machine transliteration model (Lee *et al*. 2003), finite state transducers (Knight & Grahel, 1998), and rule-based approach (Oh & Choi, 2002; Wan & Verspoor, 1998). The phoneme-based approach has received remarkable attention in various works (Lee *et al*. 2003; Meng *et al*. 2001; Oh *et al*., 2002; Knight *et al*. 1998; Virga *et al*. 2003; Jung *et al*. 2000; Al-Onaizan & Knight, 2002).

For Indian languages, as mentioned earlier, Roman transliteration has been used to represent texts of Indian languages in English. Since it is difficult to represent the letters of Hindi using just the twenty six letters of the Roman alphabet, scholars have used varying schemes to accommodate sounds that could not be correctly indicated using appropriate Roman letters. The schemes are somewhat arbitrary in the choice of Roman letters. Sometimes, phonetic symbols are used in place of the normal Roman letters. Most phonetic symbols are

basically the letters of the Roman and Greek alphabets with special marks known as diacritic marks. Roman transliteration that makes use of diacritic marks works better for Indian languages, and some standardization has been effected in the last few decades based on the recommendation from the National Library in Calcutta ("Transliteration Principles," 2008). Roman letter assignments in this scheme are phonetically equivalent to the letters of Sanskrit or other Indian languages. The primary difficulty in data entry of the phonetic symbols is that there is no provision to input the symbols directly using the standard ASCII keyboard. Transliteration methods using only the displayable ASCII symbols do not run into this problem since the ASCII letters can be typed in directly. A special computer program, however, would be required to interpret the input string to produce the Indian languages display or printout. This is precisely what the currently popular transliteration schemes attempt. Schemes such as ITRANS (Chopde, 2001), RIT (Kanneganti & Kishore, 2008), ADHAWIN (Srinivasan, 1995), and MYLAI (KalyanaSundram, 2008), use only the standard displayable ASCII letters and symbols to transliterate the text. These schemes allow multiple representations for certain syllables and long vowels but the processing program handles this well. Although the input can be done using standard keyboards, these schemes are not suitable for searching the contents on the Internet. There is a need to evolve a case-insensitive transliteration scheme to facilitate searching on web.

For the Punjabi language, a Gurmukhi to Roman transliteration system using a transliteration scheme based on ISO: 15919 transliteration and ALA-LC has been developed at Punjabi University Patiala (Sharma, 2008).

All of these transliteration schemes are either from Roman to Indian languages or *vice-versa*. Transliteration among Indian languages is rather ignored. As the English language is one of the official languages of India, Roman transliteration is developed primarily to disseminate information in the English language. Little need is felt for transliteration among Indian languages. Malik (2006) has developed a machine transliteration system from Hindi to Urdu. The system converts Hindi and Urdu into a common language that may be ASCII encodings of Hindi and Urdu characters or the International Phonetic Alphabet (IPA) equivalents of Hindi and Urdu Phonemes. Then, Hindi and Urdu will be generated from the common language. This makes the system reversible.

A corpus based transliteration system for Shahmukhi to Gurmukhi has also been developed (Saini & Lehal, 2008). Their system uses statistical data from both Shahmukhi and Gurmukhi corpora like character, word, and n-gram frequencies. In this system, script mappings are performed for simple consonants, aspirated consonants (AC), vowels, and other diacritical marks or symbols. Next, the transliteration system is virtually divided into two phases. The first phase performs rule-based transliteration, and the second phase performs the transliteration using bi-gram language model.

A machine transliteration system has also been developed for transliterating from Hindi to Punjabi (Goyal & Lehal, 2009). This system has implemented approximately fifty complex rules for making the transliteration between Hindi-Punjabi language pair. The system claims 98% accuracy. The system is not reversible due to the dependency on language-specific rules.

As more and more data in Indian languages is available in electronic format, the need of the hour is to develop sophisticated transliteration modules for Indian languages that can be used in various NLP systems, like machine translation and cross language information retrieval. This paper discusses the first attempt to develop a transliteration system for transliterating Punjabi text to Hindi.

3. Gurmukhi and Devanagari Scripts

Punjabi can be written in Gurmukhi script or in Shahmukhi script. Shahmukhi is used to write and record the Punjabi language in West Punjab in Pakistan. In East Punjab in India, however, the Gurmukhi script is used to record the Punjabi language. This paper also considers this script for writing Punjabi. The script used for writing Hindi is called Devanagari. Both Gurmukhi and Devanagari descended from the Brahmi script of Ashoka. Both scripts are left-to-right and words are spelled phonetically. As a background for readers who are not familiar with script terminology, we will describe the basic elements of a script here. Building blocks of a script are alphabets, which are the sets of letters, each of which represents one or more phonemes in the language they are used to write. In some cases, combinations of letters are used to represent single phonemes, as in the English sh, ch, and th. The alphabets contain letters for both consonants (sound articulated with complete or partial closure of the vocal tract) and vowels (sound pronounced with an open vocal tract). A single vowel or a consonant plus a vowel make a syllable. A writing system can be syllabary or alphasyllabary. A syllabary system is a phonetic writing system consisting of symbols representing syllables. On the other hand, an alphasyllabary system consists of symbols for consonants and vowels. The alphasyllabary system is also known as abiguda. Both Gurmukhi and Devanagri are alphasyllabary in nature. Alphabets of both scripts represent syllables. All consonants contain an inherent vowel /a/ or schwa ending, both of which can be altered or muted by means of diacritics or *matra*. Vowels can also be written with separate letters when they occur at the beginning of a word or on their own. When two or more consonants occur together, special conjunct symbols are often used to add the essential parts of the first letter or letters in the sequence to the final letter (“Writing Systems,” 2010).

Gurmukhi, meaning "from the mouth of the Guru" is the most commonly used script in India for writing in Punjabi. Gurmukhi was introduced by the second Guru of the Sikhs, Guru Angad Dev Ji, in the sixteenth century as a polished version of the Landa writing system, which was used in old Punjabi at that time. Part of the shift from Landa to Gurmukhi entailed

the inclusion of some Devanagari letters. Gurmukhi was then used to record the scriptures of the Sikhs, and it continues till today as the prominent writing system for Punjabis. Bhatia (1993) identifies three stages of development the Punjabi language has undergone over the years, which are illustrated in Figure 1 below.

Stage 1 : Old Punjabi (10th to 16th century)
Stage 2 : Medieval Punjabi (16th to 19th century)
Stage 3 : Modern Punjabi (19th century to present)

Figure 1. The development stages of the Punjabi language.

In Gurmukhi, there are three letters that are used to provide bases for free-standing vowels:

ੳ [uṛa] ਅ [æṛa] ੲ [iṛi]

There are nine dependent vowel sounds (also called diacritics) available in Punjabi as ਾ, ਿ, ੀ, ੁ, ੂ, ੇ, ੈ, ੋ, and ੌ. The allowed combination of vowel symbols and sounds are ਅ, ਆ, ਇ, ਈ, ਉ, ਊ, ਏ, ਐ, ਓ, and ਔ. These are also called independent vowels. All of these combinations have a unique code in Unicode and are termed as independent vowels. When a vowel sound comes at the start of a word or is independent of some consonant in the middle or end of a word, independent vowels are used. There are 32 consonants in the older Gurmukhi script. Later, some more letters were introduced to represent some sounds adapted from Farsi and Persian. Presently, Gurmukhi has 38 consonants, 10 vowels letters (independent vowels), 9 vowel symbols (dependent vowels), 2 symbols for nasal sounds, and 1 symbol that duplicates the sound of consonants (Bhatia, 1993; Malik, 2006). The form of each syllable is often dependent on whether it occurs in combination with other letters, *e.g.*, a vowel following a consonant changes its form, behaving as a diacritic modifier to the consonant:

ਕ + ੀ = ਕੀ

All consonants have an inherent vowel, *i.e.* /a/. Diacritics, which can appear above, below, before or after the consonant they belong to, are used to change the inherent vowel. When certain consonants occur together, a special conjunct symbol *halant* character ੍ before the consonant is used to combine the essential parts of each letter. For example

ਕ + ੍ + ਰ = ਕ੍ਰ

These are also called conjunct consonants. A sentence illustrating Gurmukhi script is given below. Terminology used throughout the paper is: TT denotes phonetic transcription, G denotes gloss, and E denotes English sentence.

“ਪੰਜਾਬੀ ਮੇਰੀ ਮਾਂ ਬੋਲੀ ਹੈ।”

TT: pañjābī mērī māṃ bōlī hai.

G: Punjabi my mother tongue is.

E: Punjabi is my mother tongue.

The Nāgarī (lit. 'of the city') or Devanāgarī ('divine Nagari') was originally developed to write Sanskrit but was later adapted to write many other languages, including Hindi. Devanagari has 65 consonants, 18 full vowel letters, 17 vowel symbols, and 2 symbols for nasal sounds. Hindi uses only 34 consonantal syllables, 11 vowel letters (independent vowels), 9 vowel symbols (dependent vowels), and 2 symbols for nasal sounds (“The Devanagari Script Block,” 2008). Similar to Gurmukhi, the form of each syllable is often dependent on whether it occurs in combination with other letters, *e.g.*, a vowel following a consonant changes its form, behaving as a diacritic modifier to the consonant:

क [ka] + आ [ā] = का [kā]

When two consonants occur together, the combination results in a conjunct character. The first sign receives the diacritic ् known as a *halant*, to show that the consonant has its inherent vowel silenced, also known as the *dead consonant form*.

क् [ka] + र [ra] = क्र

Unlike Devanagari, there are only a few conjunct consonants in Gurmukhi. In Devanagari, if a consonant is followed by two vowels, then the first vowel is written in the diacritic form, whereas the second is written in full. This feature is not present in Gurmukhi. Vowels may also be nasalized, which is indicated by a anusvara “ ं” or *chandrabindu* “ ँ” written over the head stroke (horizontal line) of the vowel or the consonant it is attached to.

हुँ [hũ] nasalized

In cases where part of the vowel is written above the head stroke, only the dot is used to indicate nasalization. Although chandrabindu is not a part of Standard Hindi, in practice, both are used interchangeably in Hindi. The sentence illustrating Hindi is given below:

“पंजाबी मेरी मातृभाषा है।”

paṁjābī merī mātṛbhāṣā hai

G: Punjabi my mother tongue is.

E: Punjabi is my mother tongue.

Except for minor differences, most of the letters are same in both of the scripts. There are three syllables of consonant clusters in Hindi that are written as a single atomic grapheme, *i.e.*, त्र , ज्ञ, क्ष but no such letters or consonant clusters are available in Gurmukhi. Punctuation in

Punjabi is similar to Hindi. Table 1, adapted from (Goyal & Lehal, 2009), shows the similarities and dissimilarities among alphabets in both scripts. As we can see, most of the letters have one to one correspondence, so this table also forms the base for direct mapping.

Table 1. Alphabet of Gurmukhi and Devanagari scripts

Gurmukhi	Devanagari	Gurmukhi	Devanagari	Gurmukhi	Devanagari	Gurmukhi	Devanagari	Gurmukhi	Devanagari	Gurmukhi	Devanagari	Gurmukhi	Devanagari
ੳ	-	◌ਾ	◌ा	ਕ	क	ਟ	ट	ਨ਼	ऩ	ਲ	ल	ਗ਼	ग़
ਅ	अ	◌ਿ	◌ि	ਖ	ख	ਠ	ठ	ਪ	प	ਲ਼	ळ	ਜ਼	ज़
ੲ	-	◌ੀ	◌ी	ਗ	ग	ਡ	ड	ਫ਼	फ़	-	ऴ	ੜ	ड़
ਆ	आ	◌ੁ	◌ु	ਘ	घ	ਢ	ढ	ਬ	ब	ਵ	व	ਫ	ढ़
ਇ	इ	◌ੂ	◌ू	ਙ	ङ	ਣ	ण	ਭ	भ	ਸ਼	श	ਭ	फ़
ਈ	ई	◌े	◌े	ਚ	च	ਤ	त	ਮ	म	-	ष	ਯ਼	य़
ਉ	उ	◌ੈ	◌ै	ਛ	छ	ਥ	थ	ਯ	य	ਸ	स	ੜ	ऱ
ਊ	ऊ	◌ੋ	◌ो	ਜ	ज	ਦ	द	ਰ	र	ਹ	ह	-	ऩ
ਏ	ए	◌ੌ	◌ौ	ਝ	झ	ਧ	ध	ਰ਼	ऱ	ਕ਼	क़	ਹ	ह़
ਐ	ऐ	-	◌ॄ	ਞ	ञ	ਨ	न	-	ऱ	ਖ਼	ख़	ਵ	व़
ਓ	ओ	-	◌ॢ	-	◌ृ								
ਔ	औ	-	◌ॣ	ੰ	-								

4. Approach to Transliteration from Gurmukhi to Devanagari

4.1 Letter to Letter Mapping

Both Punjabi and Hindi are phonetic languages, and their scripts represent the phonetic repository of their respective languages. These phonetic sounds are used to determine the relations between the alphabets of the two scripts. On the basis of this idea, character mappings are determined. The development of the Unicode system for representing alphabets eases the problem to some extent. With this system, every letter can be uniquely mapped to the corresponding letter, as shown in Table 1. Taking into account the similarity of both of the scripts, letter to letter mapping is the obvious choice for the baseline computation. Letters are mapped using Table 1. This system is used as the baseline for improvements by the rule based

and Soundex based approaches.

For analysis and comparison purposes, both scripts are subdivided into different groups on the basis of types of letters, *e.g.*, consonants, vowel symbols, vowel sounds, and other symbols.

4.1.1 Vowel Mapping

Punjabi contains 10 vowel symbols. Hindi vowels have one to one correspondence with Punjabi vowel symbols. There are nine dependent vowel sounds available in Punjabi as ਾ, ਿ, ੀ, ੁ, ੂ, ੇ, ੈ, ੋ, and ੌ. Corresponding vowel sounds in Hindi are ा, ि, ी, ु, ू, े, ै, ो, and ौ. When a vowel sound comes at the start of a word or is independent of any consonant in the middle or the end of the word, independent vowels are used. The mapping is shown in Table 2.

Table 2. Vowel Mapping

Independent vowels		Dependent vowels	
Gurmukhi	Devanagari	Gurmukhi	Devanagari
ਅ	अ	ਾ	ा,
ਆ	आ	ਿ	ि,
ਇ	इ	ੀ	ी,
ਈ	ई	ੁ	ु,
ਉ	उ	ੂ	ू,
ਊ	ऊ	ੇ	े,
ਏ	ए	ੈ	ै,
ਐ	ऐ	ੋ	ो,
ਓ	ओ	ੌ	ौ
ਔ	औ		

Besides these vowels, Punjabi has two more symbols, ੳ and ੲ. No symbol is present in Hindi for these two symbols. The nearest phonetic letters are उ and इ, respectively. These are used in the letter to letter mapping scheme.

4.1.2 Consonant Mapping

Consonant mapping is shown in Table 1, Columns 5-14. No letter in Punjabi is present for the Hindi letters "क़", "ष", and "स़". This means these letters can never be mapped in a letter to letter based approach. Similar is the case for some double-sound-producing syllables, like "क्ष", "ज्ञ", and "श्र". Syllables like "ज्ञ" and "श्र" can be mapped by a rule based approach while others like "ष", "क्ष", "क़", and "स़" can be handled by the Soundex based approach discussed in the next sections.

4.1.3 Conjunct Consonants Mapping

There are three conjunct consonants in Gurmukhi also called PAIREEN, ੍ਹ ("*Haahaa*"), ੍ਰ ("*Raaraa"*), and ੍ਵ ("*Vaavaa*"), which are shown in Table 3.

Table 3. Sub Joins of Gurmukhi

Conjunct Consonants	Punjabi	Hindi	English
੍ਹ	ਬੁਲ੍ਹੜ [bulharh]	बुल्हड़	A person with large lips
੍ਰ	ਪ੍ਰੀਤਮ [pritam]	प्रीतम	A person name (Pritam)
੍ਵ	ਸ੍ਵਰਗ [svarag]	स्वरग	Heaven

The usage of PAIREEN *"Haahaa"* is quite frequent in Punjabi, but the other two are very rare in usage and are used in Sanskrit loan words. In Punjabi, they are represented by the *halant* character ੍ before the consonant, which indicates that the inherent vowel is omitted. A similar *halant* character is also present in Hindi, which can replace the *halant* character in Punjabi. It may be noted that the actual sequence of letters is the same in both languages but the visual sequence of letters differs (Table 4). PAIREEN *"Haahaa"* in Punjabi is replaced with their full consonant counterparts, while the full consonant preceding them in Punjabi is shown in half form in Hindi. Similar is the case with PAIREEN "Vaavaa". On the other hand, PAIREEN "*Raaraa"* takes the position below the previous consonant in Punjabi as well as in Hindi.

Table 4. Actual Sequence of letters

ਬ	ੁ	ਲ	੍	ਹ	ੜ	=	ਬੁਲ੍ਹੜ
ब	ु	ल	्	ह	ड़	=	बुल्हड़

4.1.4 Other Symbols

Punctuation marks and digits are the same in both scripts. A special character called as visarga (ः) is present in Hindi but not in Punjabi. So, it will never be mapped in the letter to letter

based scheme. Besides this, Gurmukhi has two separate nasal characters, Bindi (ਂ) and Tippi (ਁ). Hindi has only one nasal character called anusvara " ं". In the Devanagari script, anusvara is represented with a dot (*bindu*) above the letter *e.g.* "मं". Both nasal characters of Punjabi are mapped to this single nasal character in Hindi. Adhak (ੱ) is used to duplicate the sound of a consonant in Punjabi. No such character is present in Hindi. Sound duplication is represented by half form consonants in Hindi.

4.1.5 Problems in Letter to Letter Mapping

The total number of letters in Punjabi and Hindi are not same (Table 1). The Punjabi letters ੳ and ੲ have no mapping in Hindi. Similarly, there are letters in Hindi that have no mapping in Punjabi *e.g.* ऋ. These letters will never be mapped in Punjabi to Hindi transliteration using a direct mapping method. Some letters have more than one representation in Hindi, *e.g.,* श may be mapped to श or ष. Another problem is the use of conjunct consonant forms in Hindi. In Hindi, a syllable may consist of a vowel, a consonant followed by vowel, or a consonant cluster followed by a vowel. The last form *i.e.,* when two or more consonants are used within a word with no intervening vowel sound, is known as a conjunct consonant. Use of conjunct consonants is limited in Punjabi. Only three letters can be used as conjuncts *i.e.*, ਹ, ਰ, and ਵ. Their representation is also unique. It is not a trivial task to find out which combinations of letters in Punjabi will take conjunct consonant form in Hindi. For example, why the word ਨਿਊ [niū] (new) in Punjabi takes the conjunct consonant form in Hindi न्यू, is not clear.

Also, the mapping of nasal consonants is not clear. Nasal consonants in initial place in a conjunct may be expressed using the anusvara over the previous vowel, rather than as a half-glyph attached to the following consonant. The anusvara is written above the headstroke, at the right-hand end of the preceding character. In the list below, both spellings are correct and equivalent, although anusvara is preferred in the case of the first two: रंग = रङ्ग, पंजाबी = पञ्जाबी, हिंदी = हिन्दी, लंबा = लम्बा. Anusvara is still applied when previous character has its own vowel sign. If the vowel sign is [aa], the anusvara appears over the [aa], *e.g.* फ़ांसीसी or आंदोलन.

Similarly, the position of the character "र" is not specific. It can be used as consonant, subscripted consonant, or can take a position above a consonant or diacritics, and it is typically displayed as a small mark above the *right* shoulder of the last letter in the syllable. It is not clear how the character ਰ [ra] in Punjabi will map to which form in Hindi, *e.g.*, the three cases where mapping of ਰ is not clear are: from ਤਰਕਸ਼ [tarkash] (arrow-holder) to तरकश, from ਵਰਤ [varat] (fast) to व्रत, and from ਬਰਤਨ[bartan] (utensil) to बर्तन.

4.2 Rule Based Approach

Character mapping alone is not sufficient for a Punjabi to Hindi machine transliteration system. Quite a reasonable improvement can be achieved by small amount of dependency or contextual rules. These rules are manually crafted with the help of a linguist by observing the problems in the direct mapping system. This section discusses such rules for alleviating some of the problems discussed in the previous section. Ideally, the rule set must contain all of the rules required by the system to perform, but practically it is not possible to construct such a set. About 11 rules are identified that are applicable on the source text and about 8 rules are identified that are applicable on the target text. We tried to cover all the possible cases but still we may miss some of them. Moreover, new rules can be added when identified. The rules go as follows:

1. Letters whose mapping is not available in Hindi :
   a. ੳ and ੲ are two such letters whose corresponding Hindi letters are not available. They are replaced by their most phonetically equivalent letters, *i.e.*, उ and इ respectively. Formally, the rule goes like this

      if inpt_str contain "ੳ" then replace it with "उ"

      if inpt_str contain "ੲ" then replace it with "इ"

   b. A character adhak "ੱ" is present in Punjabi and used to show the stress on the next character. No letter in Hindi is present to represent this character. Nevertheless, the purpose is served by placing a half character before the stressed character, *e.g.,* "ਸ਼ੱਕਰ" [śakkar] (jaggery) is transliterated as "शक्कर"[śakkar]. There is an exception for this rule. If the next character of adhak "ੱ" is ਖ [kha] then, instead of placing a half character, a half क[ka] is placed, *e.g.*, "ਮੱਖਣ" [makkhaṇ] (butter) transliterated to "मक्खन" [makkhana]. Also, if the next character is ਛ [ccha], then the half character is replaced by a half च, *e.g.*, as in "ਮੱਛਰ" [macchar] (mosquito), which is transliterated to "मच्छर" [macchar]. Similarly, if the next character is ਧ [dha], then the half character is replaced by a half द, *e.g.,* as "ਬੱਧ" [baddh] is transliterated as बद्ध. Formally, it is

      If current_char = "ੱ" then
        If nxt_char= "ਖ" then
            map "ੱ" to "क्"
        else if nxt_char= "ਛ" then
            map "ੱ" to "च्"
        else if nxt_char= "ਧ" then
            map "ੱ" to "द्"
        else
            map "ੱ" to nxt_char & "्"
        end if
      end if

2. Bindi and Tippi both are used to produce a nasalized sound in Punjabi. There is only Bindi (anusvara) present in Hindi to serve the same purpose. Tippi can be successfully mapped to Bindi in Hindi. A special case is when the tippi " ◌ੰ" [ṃ] is followed by "ਨ" [na] or "ਦ" [da], then it is replaced by a half "न", as in "ਕੰਨੜ" [kannaṛ] (Kannad) "कन्नड़" [kannaṛ]. If, however, it is followed by "ਬ" [ba], then it is replaced by a half "म", as in "ਮੋਜ਼ੰਬਿਕ" [mōzmbik] (mozambique) "मोज़म्बिक". Formally, it is

```
If current_char = "◌ੰ" then
  If nxt_char= "ਨ" then
        map "◌ੰ" to "न्"
  else if nxt_char= "ਬ" then
        map "◌ੰ" to "म्"
  else
        map "◌ੰ" to "◌ं"
  end if
end if
```

3. The presence or absence of the diacrtic mark "ਿ" in the target string is effected by the character ਹ [ha] in Punjabi. When ਹ [ha] in the input string is followed or preceded by the ਿ [i] mark, ਿ [i] is translitrated differently. In this case, ਹ [ha] is followed by ਿ then ਿ is ommited from the transliterated text, *e.g.*, ਸ਼ਹਿਰ [shahir] (city) is transliterated to शहर. On the other hand, if ਹ [ha] is preceded by ਿ, then ਿ is mapped to े in the transliterated text, *e.g.*, ਸਿਹਤ [sihat] (health) is transliterated to सेहत. Formally, it is

```
If current_char = "ਹ " then
  If nxt_char= "ਿ" then
        Omit "ਿ" in target string
  else if prev_char= "ਿ" then
        map "ਿ"  to "े"
  else
        map "ਿ" to "ਿ"
  end if
end if
```

4. Another rule deals with transliteration of letters ਅਰ [ar]. If this combination appears at the final position in a word, then, instead of mapping ਅ [a]to अ, this letter is mapped to य., *e.g.*, ਬੀਅਰ [bīar] (bear) is transliterated to बीयर. Formally, it is

If last_two_char = "ਅਰ " then

map "ਅ" to "य"

else

map "ਅ" to "अ"

end if

5. If ਅ is in the last position, then it is replaced by य as in ਪ੍ਰਿਅ [pria] – प्रिय Formally, it is

   If last_char = "ਅ " then map "ਅ" to "य"

6. If ਈ is in the last position, then it is replaced by यी, *e.g.*, in ਵਾਜਪਾਈ [vājapāī] (Vajpai)– वाजपायी Formally, it is

   If last_char = "ਈ " then map "ਈ" to "यी"

7. If ਓ is in the last position or the 2nd to last position, then it is replaced by यो, *e.g.*, in ਗਲੀਲਿਓ [galīliō] (galilio) – गलीलियो. Formally, it is

   If last_char = "ਓ" or second_last_char= "ਓ" then map "ਓ" to "यो"

8. A rule for plural forms of Punjabi words calls for changing the vowel sign of the last chararcter. It says that, if the last characters are ◌ਾ ◌ੰ, then ◌ਾ is mapped to ◌ੋ in Hindi, *e.g.*, ਕਿਤਾਬਾਂ [kitābāṃ] (books) is transliterated to किताबों[kitābõ]. Although this is a case of translation where the root ਕਿਤਾਬ [kitāb] is the same in the two languages and gets inflected by the rules in respective languages, this case can be handled at the transliteration stage. Formally, it is

   If last_two_char = "◌ਾ ◌ੰ" then map "◌ਾ" to "◌ो"

9. Similarly, if the last positions are occupied by ਆਂ [āṁ], then ਅ [a] is mapped to य, *e.g.*, ਸ਼ੀਸ਼ੀਆਂ [shīshīāṃ] (bottles) should be mapped to शीशीयों rather than शीशीओं. It is worth mentioning here that both of the spelling variants are acceptable in general to Hindi speakers but the former is preferred. Formally, it is:

   If last_two_char = "ਆਂ" then map "ਅ" to "य"

10. The character ੴ [ik-ōṅkār] is replaced by string "एक ओंकार". Formally,

    If char = "ੴ" then map "ੴ" to "एक ओंकार"

11. The letter "ਣ" [ṇ] is converted to "न" if it is a last consonant in a word, *e.g.*, in ਕਹਿਣਾ [kahiṇā] (to say) the letter ਣ is converted to न when transliterated to produce "कहना".

    If last_consnt = "ਣ" then map "ਣ" to "न"

The above rules are applicable for the character sequence found in the source script. Table 5 describes some rules that are applicable to the character sequence found in the target script, *i.e.* Hindi. We can find a probable character sequence using the letter to letter mapping approach. These rules then can be applied to these probable strings. Every row of Table 5 gives the

replacement rules for a character combination found in the probable string. For example, for a source script word ਵਿਉ [viu] (view), the letter to letter mapping approach produces a probable string विउ, which is converted to व्यु by applying the rule from the following table. Formally, it can be described as follows:

For each chr_seqnce in table

If chr_ seqnce present in input string then

Replace chr_ seqnce to corresponding seqnce from table

Table 5. Replacements in transliterated text

Occurrence in hindi token	Replacement in hindi token	Example
"ਿਉ"	"੍ਯੁ"	ਵਿਉ [viu]–विउ – व्यु
"ਿਊ"	"੍ਯੂ"	ਨਿਊ [niū]– निऊ – न्यू
"ਿਓ"	"੍ਯੋ"	ਬਿਓਰਾ [biōrā]– बिओरा – ब्योरा
"ੀਆ"	"ਿਯਾ"	ਕੋਰੀਆ [kōrīā]– कोरीआ – कोरिया
"ੀਅ"	"ਿਯ"	ਕੋਰੀਅਨ [kōrīan] — कोरीअन — कोरियन
"�इਣ"	"ਯਣ"	ਰਾਮਾਇਣ [rāmāiṇ] — रामाइण — रामायण
"ੀਯੋ"	"੍ਯੋ"	ਟੋਕੀਓ [ṭōkīō] — टोकीयो – टोक्यो
"ਿਆ"	"੍ਯਾ"	ਸਿਆਚਿਨ[siācin]— सिआचिन — स्याचिन

4.3 Soundex Based Approach

Considerable improvements are noticed over the letter to letter mapping approach when using the rule based approach, but still some problems persist. Although we tried to cover the maximum number of rules, still there are cases for which no rule applies. Letters or syllables not available in Punjabi but present in Hindi, ऋ, क्ष ,ज्ञ ,श्र ,ष, *etc*. are still unhandled. The syllables having double tones in Hindi, like ऋ is a combination of sound of consonant "र" /r/ and vowel "ि" /i/, are represented in Punjabi with the help of two letters, like the consonant "ਰ" /r/ and vowel "ਿ"/i/. There is no means to find when these two letters in Punjabi will map to corresponding letters "रि" or to a single syllable "ऋ" in Hindi, *e.g.,* consider following cases.

Case I: ਰਿਕਸ਼ਾ(riksha) [rickshaw] → रिक्शा

Case II: ਰਿਸ਼ੀ(rishi) [cleric] → ऋषि

In Case I, the letters ਰਿ are mapped to रि, whereas, in Case II, they are mapped to ऋ. Also, the rule based approach is deficient in producing the half form of letters and some other letters, as discussed in the previous section. For generating these characters, the Soundex technique is employed. Soundex is a phonetic matching technique. The Soundex algorithm was designed by Odell and Russell (1918) to find spelling variation of names. It represents classes of sounds that can be lumped together. Two names match if they have the same Soundex representation.

Back in 1918, Odell and Russell proposed Soundex, the first phonetic encoding for English to be used in the US census. Soundex partitions the set of letters into seven disjoint sets, assuming that the letters in the same set have similar sounds. Each of these sets is given a unique key, except for the set containing the vowels and the letters h, w, and y, which are considered to be silent and are not taken into account during encoding. For example, both *realize* and *realise* have been encoded to 'R-420' in Soundex encoding, which works well for trivial cases, but fails to give same code to words where letters change its pronunciation in different contexts. For example, *knight*, *night,* and *nite* are similar sounding words but Soundex does not give the same code to these words.

For Punjabi to Hindi transliteration, the Soundex concept is extended for searching for the correct spelling variant of a given transliteration. The Soundex codes are assigned to the Hindi characters based upon their phonetics *i.e.,* similar sounding characters get the same code, *e.g.,* both श and ष have the same sound, thus, they get the same code, 44. The akharas having single graphemes get the code according to the sounds they produce, *e.g.,* ऋ is a combination of sound of consonant "र" /r/ and vowel "ि" /i/, so it gets the code of these two letters, *i.e.* 4149. The code table is shown in Table 6. Codes in this table are selected in such a way that similar sounding letters get the same code. We start with independent vowel sounds whose code ranges from 1 to 5. For consonants, we start from two digit codes, *i.e.* 11 to 47. Thereafter, we code the dependent vowels and some other miscellaneous letters. Unlike Soundex code, which consists of a letter (the first letter of the string) followed by three numerical digits encoding the remaining consonants, all characters of the Hindi string are coded into digits. This ensures matching in case the first letter does not match. From the transliterations produced by the previously discussed methods, the correct spelling variant is searched using Soundex codes as described in the following lines. Let *H* be a set of equivalent Hindi words and *TH* be a transliteration produced by the machine transliteration step. A relevant *h* from *H* is selected by comparing phonetic similarity between *H* and *TH*. For implementation purposes, we make a unigram table from a corpus of about 1 GB size obtained from the Internet. Unigram contains about 150,000 unique words along with their frequency in the corpus. The codes are generated for each of the word in unigram using Table 6. For the comparison, letters in *TH* are converted into phonetic code using the mapping table as described in Table 6. Then, this code is looked for in a unigram table. The unigram table may

produce more than one candidate. The candidate with maximum frequency is selected as the correct variant of the given *TH* produced by the machine. For example, consider the word "ਡਰਾਫਟ" [ḍarāphaṭ] (draft) written in Punjabi. The string "डराफट" is produced by the baseline module. The code 2541483623 is generated for this string, as shown in the following table.

ड	र	ा	फ	ट
25	41	48	36	23

This code is looked for in a unigram database. This database contains two entries against this code, *i.e.* ड्राफ्ट and डराफ्ट with frequency 12 and 8. The string with higher frequency is selected as the correct output for this input.

Table 6. The coding scheme for Hindi text

अ	1	च	17	न	34	ा	48
ऑ	1	छ	18	प	35	ॉ	48
इ	2	ज	19	फ	36	ि	49
ई	2	ज़	19	फ़	36	ी	49
उ	3	ज्ञ	20	ब	37	ो	50
ऊ	3	झ	21	भ	38	ौ	50
ए	4	ञ	22	म	39	ू	51
ऐ	4	ट	23	य	40	ु	51
ओ	5	ठ	24	र	41	े	52
औ	5	ड	25	ऱ	42	ै	52
क	11	ड़	26	ल	42	ँ or ं	53
क़	11	ढ	27	ळ	42	़	54
क्ष	12	ढ़	28	व	43	ॐ	55
ख	13	ण	29	श	44	ॠ	4149
ख़	13	त	30	ष	44	ॠ	4149
ग	14	थ	31	आ	45	ृ or ॄ	4149
ग़	14	द	32	स	46	श्र	4441
घ	15	ध	33	ह	47		
ङ	16						

5. The Experiment

In this section, we describe the evaluation of our models on the task of Punjabi to Hindi transliteration.

5.1 Data

A provocative question for evaluators is the type of test material that should be adopted for evaluation. Balkan (1994) describes three types of test material, *viz*. test corpora, test suites, and test collections. While test corpora is a collection of naturally occurring texts, a test suite is a collection of artificially constructed inputs, where each input is designed to probe a system's treatment of a specific phenomenon or set of phenomena. A test collection is a set of inputs associated with a corresponding set of expected outputs. Test suites and test corpora are extensively used for evaluating various systems involving NLP, while test collections are rarely used.

Users can make their own test suites, and such test suites are helpful for checking the improvement of an MT system. For two competing systems, however, test suites are not considered a good method of assessment. It is difficult to interpret the result that both systems transliterate same percentage of text but fail on different sentences. How to design a test suite, the cost of constructing it, and what sort of sentences/words should go into a test suite are some other issues discussed by Arnold (1995). There are several issues involved in the selection of a set of words for a comprehensive evaluation. For example, the set could be constant, variable, or a mixed one; the collection of words may be domain-specific or generic. It is obvious that there is no guarantee that even the bulkiest sample will include all of the possible words of the source language. Some suggestions as to what should go into a test suite are discussed in Hoard (1991). Some empirical evidence is present in literature to answer the question of how much text to include in a test suite. Although this evidence is not for transliteration system, it can provide insight for this system as well. Elliott *et al*. (2003) tried to prove the general hypothesis that more text would lead to more reliable scores. Based on an empirical assessment of score variation, the authors estimate that systems could be reliably ranked with test suite of around 14,000 words. Zhang & Vogel (2004) also studied the influence of the amount of test data on the reliability of automatic metrics, focusing on confidence intervals for BLEU and NIST scores. Their results show that BLEU and NIST scores become stable when using around 40% of the data (around 40 documents or 300 sentences), although stability is defined here in terms of the distance between scores of different systems. Estrella *et al*. (2007) shows that, for human or automatic evaluation, about five documents from the same domain-with 6,000 words-seem sufficient to establish the ranking of the systems and about ten documents are sufficient to obtain reliable scores.

For this experiment, a benchmark sampling method is used to select the set of words. Unicode encoded text is used in Punjabi as well as Hindi, as it eliminates the need of normalizing the text. If text is in font encoded form, then the ASCII code for a given character may vary. For example, under the AnmolLipi font, the Latin character 'A' would appear as 'ਅ'. Conversely, under the DrChatrikWeb font, it would appear as 'ੳ'. Unicode encoded text provides a unique way of representing a character; hence, it eases the task of mapping letters from different alphabets. We tested on people names, location names, and foreign words in Punjabi. The list was prepared manually by typing the names and terms from telephone directories, census records, *etc*. and partially obtained from manually searching the Internet. We tried to include the maximum possible variety of words in the set. The words are further categorized as Punjabi names, Hindi names, English names, and other foreign names/words. It is worth mentioning here that quite a number of English words are adapted into the Punjabi and Hindi languages, so we categorized them separately from the foreign words, which are words from other languages. The list contains about 3500 person names, 1500 location names, and 1000 foreign words.

5.2 Evaluation Methodology

The following combinations of approaches are tested for Punjabi-Hindi transliteration task.

Baseline: As a baseline for our experiments, we used a simple letter to letter based approach that maps Punjabi letters to the most likely letter in Hindi. We call it CASE-I.

Baseline followed by Rule Based Approach: Some hand crafted rules are studied for the improvements in the letter to letter based mapping system. This system is called CASE-II.

Baseline followed by Soundex Approach: The Soundex approach is applied to the output of the baseline method. This is termed as CASE-III.

Baseline followed by Rule Based followed by Soundex Based Approach: Both the rule based and Soundex approaches, consecutively, are applied to the output of the baseline method. This is termed as CASE-IV.

Baseline followed by Soundex Based followed by Rule Based Approach: Both the Soundex based and Rule based approach, consecutively, are applied to the output of the baseline method. This is termed as CASE-V.

Human: For the purpose of comparison, we allowed an independent human subject (fluent in Punjabi and native speaker of Hindi) to perform the same task. The subject was asked to transliterate the Punjabi words in the test set without any additional context. No additional resources or collaboration were allowed. This output was used as the gold standard for checking the system's performance.

5.3 Evaluation Metrics

Often, more than one transliteration is acceptable for any input string. Using one gold standard may prove to be too rigid for measuring accuracy. On the other hand, including more than one correct transliteration complicates the computation of the evaluation score. Also, there is no one exact transliteration of a foreign word, so a soft standard is required to indicate the closeness of output to the gold standard. Levenshtein distance is used for this purpose. The metrics are described as follows:

Word Accuracy Rate: It is defined as percentage of correct transliteration from the total generated transliterations by the system.

Average Levenshtein Distance: This is the average of Levenshtein distances between the transliterated word and reference word. Levenshtein distance is a measure of similarity of two strings. The distance is the number of deletions, insertions, or substitutions required to transform the source string to the target string, *e.g.,* transforming from the source string इखतिआर, which results from the direct transliteration of ਇਖਤਿਆਰ [ikhtiār], the correct target string इख्त्यार can be produced by deleting ि from त, adding halant (्) to त, and replacing अ with य. Thus, the Levenshtein distance is 3.

While Levenshtein distance captures the performance of a system at character level, word accuracy rate is used to capture the performance at word level. A word may have non-zero Levenshtein distance but still may be accurate. Lower Levenshtein distance value implies that, although the result is not the same as that of the gold standard, it is still intelligible.

6. Evaluation Results

The experiment was performed on the prepared list. The results of this experiment follow.

Table 7. Word Accuracy Rate and Avg Lev Dist. of Fivr Cases

	CASE-I	CASE-II	CASE-III	CASE-IV	CASE-V
WAR	73.13%	78.88%	86.72%	92.65%	87.15%
ALD	0.61	0.32	0.39	0.10	0.37

The baseline model produced a 73.13% accuracy rate. The rule based enhancement and Soundex based enhancement shows the improvements in performance with Soundex prove to be better than rule based. It is observed that CASE-V, *i.e.* Soundex followed by rule based, does not provide much improvement. This is because in the literature, the Soundex approach is largely discussed as a spelling correction method. Thus, it performs better on refined text, *i.e.* the output generated by the rule base, and fails to perform on raw input. Similarly, the average edit distance of different cases is shown in Table 7. The average edit distance of

CASE-III is increased from CASE-II then decreased with WAR increasing, as evident from Figure 2. This is because the human evaluator has marked some spelling variations correct. That is, for the name ਸੁਰਿੰਦਰ (*Surinder*), the three transliterations produced by CASE-I, Case-II, and Case-III are सुरिंदर, सुरिन्दर and सुरिंद्र, respectively. The human evaluator marked all of these as correct. Nevertheless, as Levenshtein distance is measured against a gold standard, which is सुरिंद्र in this case, CASE-III shows an increase in this parameter despite increased word accuracy. The word accuracy rate for CASE IV, *i.e.* the Base + rule + Soundex approach is found out to be 92.65% and Avg. Levs. Dist is 0.10. These figures are the highest among all the cases and verify the suitability of the approach.

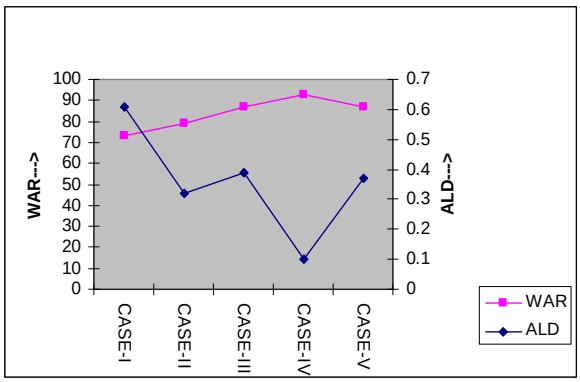

Figure 2. Word Accuracy Rate and Avg Lev Dist. of Five Cases

The breakdown of the figures is shown in following tables.

Table 8. WAR and ALD for person names, location names, and foreign words in all cases

	Person Name		Location Name		Foreign Words	
	WAR (%)	ALD	WAR (%)	ALD	WAR (%)	ALD
CASE-I	75.85	0.59	67.10	0.61	63.50	0.64
CASE-II	86.9	0.23	79.8	0.24	69.8	0.50
CASE-III	88.5	0.41	81.7	0.51	89.8	0.26
CASE-IV	92.8	0.1	91.45	0.1	93.8	0.09
CASE-V	88.78	0.39	82.7	0.48	90.01	0.24

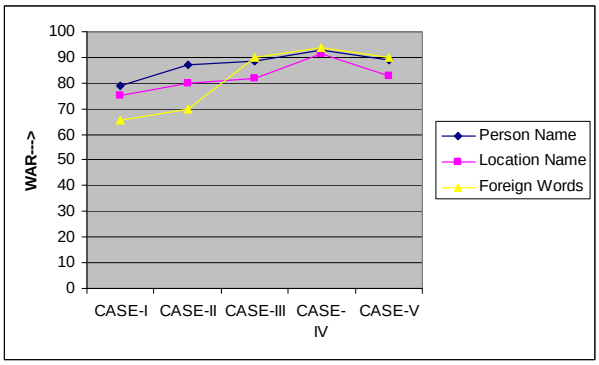

Figure 3. WAR for person names, location names, and foreign words in all cases

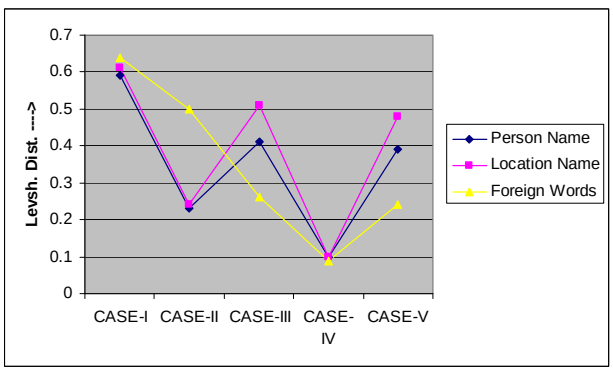

Figure 4. ALD for person names, location names, and foreign words in all cases

The word accuracy for words whose origin is also Punjabi is 86.6 in the baseline model and shows the similar trend in other cases, as shown in Table 9. The improvement in all cases is registered with maximum for Hindi and English languages.

Table 9. Word Accuracy rate and Avg. Lev Dist according to origin of source language of input word

	Punjabi Word		Hindi Word		English Word		Foreign Word	
	WAR	ALD	WAR	ALD	WAR	ALD	WAR	ALD
CASE-I	86.6	0.41	63.54	0.76	57.3	0.83	80.1	0.39
CASE-II	91.6	0.12	70.15	0.44	63.4	0.60	83.9	0.25
CASE-III	90.1	0.36	83.29	0.46	85.3	0.35	90.05	0.21
CASE-IV	95.4	0.06	89.9	0.14	92.2	0.10	93.37	0.08

The accuracy in the baseline model for Hindi and English is quite low because the baseline model cannot capture the half form representation of letters. As there is no half form in

Punjabi, a Hindi name, when spelled in Punjabi, uses the full form of the character instead of its half form. For example, the name इन्द्र (Inder) when spelled in Punjabi will look like ਇੰਦਰ [indar]. Here, the half form of न and र in the Hindi name are represented by (tippi-a character for nasal sound) and ਰ, respectively. When this form is transliterated by the baseline model, it will produce इंदर, which is incorrect. Similar is the case with English and other foreign words. So, due to the character gap in Punjabi and other languages, the word accuracy rate for the baseline is low. It is also interesting to note that when words that are originally from Hindi and used in Punjabi are transliterated back to Hindi, the accuracy rate is lower than other types of words (for Cases III and IV). The reasons are twofold. First, Hindi words written in Punjabi are an approximate transliteration of the original Hindi word in Punjabi. Depending upon the perception of the transliterator, a Hindi word may have more than one representation in Punjabi, *e.g.*, the Hindi name आचार्या (*Acharya*) can be represented in Punjabi as "ਆਚਾਰਿਆ ", " ਆਚਾਰੀਆ ", " ਅਚਾਰੀਆ ", and " ਅਚਾਰਿਆ ". Only the first representation, when again transliterated back into Hindi, converts to the correct representation. Other representations are converted into such strings by baseline and rule based modules that they have no match in the unigram table, making the Soundex module inefficient. Normalization of spellings at the source by a similar Soundex approach may improve the results.

Table 10. Breakdown of Word Accuracy rate and Avg. Lev Dist for person names, location names, and other words according to origin of source language of input word

		Punjabi Words		Hindi Words		English Words		Foreign Words	
		WAR	ALD	WAR	ALD	WAR	ALD	WAR	ALD
CASE-I	Person Name	86.9	0.4	63.92	0.77	79.51	0.59	82.78	0.39
	Location Name	76.61	0.60	67.09	0.48	42.30	1.23	73.68	0.26
	Other Words	87.02	0.27	30	1.3	52.21	0.83	66.66	0.55
CASE-II	Person Name	91.6	0.12	70.49	0.43	84.33	0.32	87.41	0.24
	Location Name	91.12	0.14	75.95	0.34	57.69	0.61	73.68	0.26
	Other Words	91.8	0.11	29.04	1.25	56.63	0.71	66.66	0.33
CASE-III	Person Name	91.4	0.35	83.14	0.14	90.36	0.34	90.72	0.23
	Location Name	77.41	0.56	82.28	0.29	65.38	0.90	94.73	0.05
	Other Words	92.30	0.22	90.1	0.3	87.61	0.23	88.88	0.33
CASE-IV	Person Name	95.8	0.05	89.7	0.12	95.18	0.08	94.7	0.08

	Location Name	92.74	0.08	91.14	0.14	86.53	0.11	94.73	0.05
	Other Words	95.41	0.05	90.8	0.25	92.47	0.10	88.88	0.11
CASE-V	Person Name	91.4	0.32	85.14	0.15	88.36	0.33	89.72	0.21
	Location Name	78.1	0.54	82.8	0.28	67.38	0.85	94.23	0.05
	Other Words	92.6	0.21	91.1	0.31	86.94	0.21	89.28	0.3

The system also benefits from the spelling correction capability of the Soundex method. It can improve the results by selecting the correct output for a given wrong input, *e.g.,* for the input ਅਨੁਪਮ (correct is ਅਨੁਪਮ [anupam]), the Soundex method selects the correct variation अनुपम instead of अनूपम. On the other hand, at times it selects the wrong target word, *e.g.*, for input वैंट [kaiṇṭ] (cantt) the correct output is कैंट but the Soundex method selects केंट as output. Another problem with the Soundex based approach is that it is dependent upon the unigram table. Although this table is generated from a large amount of data, still it is not exhaustive. New terms and names are coined every day, and these need to be included in the unigram table regularly. If the data is not present in this table, then the system fails. Such cases, however, are few and overall accuracy of the system improves with the combined approach of baseline + rule based + Soundex based approach.

7. Conclusion

The results show that following the rule based followed by Soundex approach produces the best results and the words can be transliterated with considerable accuracy. A fully accurate transliteration system is not possible due to the inherent problems, missing corresponding letters in two scripts, conjunct form, *etc*. Although it is possible to transliterate across the scripts preserving the basic sounds of the source language, there will be some variations because the word in the source script may be pronounced somewhat differently in the target script, as per local conventions in each region. This results in more time required by the users to interpret the word in the context of original language from which the word comes. The transliterated text will be correctly understood only if the reader has knowledge of the conventions used for representing sounds in the source script. The aim of the module at least is to present a nearest possible transliterated text to the user instead of going blank to such words. This module is successfully used for transliterating proper nouns in Punjabi to Hindi machine translation system and in the Cross Language Information Retrieval system "PunjabiKhoj," which is a search engine for Punjabi language.

Although the system performs well, still there is room for improvement. Transcription rules based upon IPA and statistical models will be considered in the future. Testing of the

system on precision, recall, and F-score is on our future agenda.

Reference

Al-Onaizan, Y., & Knight, K. (2002). Translating named entities using monolingual and bilingual resources. *Proceedings of the 40th ACL*, Philadelphia, 400-408.

Arnold, D., Balkan, L., Humphreys, R. Lee, Meijer, S., & Sadler, L. (1995). *Machine Translation: An Introductory Guide*. NCC Blackwell, Manchester, Oxford.

Balkan, L. (1998). Test suites: some issues in their use and design. In *proceedings of the International conference* on "Machine translation: ten years on" held at Cranfield University, England, 12-14 November 1994 (Cranfield University Press, 1998).

Bhatia, T. K. (1993). Punjabi: A Cognitive-descriptive Grammar. *Descriptive Grammars,* Routledge, London.

Chopde, A. (2001). *Printing Transliterated Indian Language Documents*, ITRANS, from website http://www.aczoom.com/itrans/idoc/idoc.html (Accessed on 22 Aug 2006).

Elliott, D., Hartley, A., & Atwell, E. (2003). Rationale for a multilingual aligned corpus for machine translation evaluation. In *International Conference on Corpus Linguistics (CL2003)*, 191-200. Lancanster, UK.

Estrella, P., Hamon, O., & Popescu-Belis, A. (2007). How much data is needed for reliable MT evaluation? Using bootstrapping to study human and automatic metrics. *MT Summit XI*, 10-14 September 2007, Copenhagen, Denmark. *Proceedings*, 167-174.

Goyal, V., & Lehal, G. S. (2009). Hindi-Punjabi Machine Transliteration System (For Machine Translation System). George Ronchi Foundation Journal, Italy, 64(1), 2009.

Hoard, J. (1991). Preliminaries to the Development of Evaluation Metrics for Natural Language Semantic and Pragmatic Analysis Systems. in Neal, J.G. and Walter, S.M. (eds.), (1991): *Natural Language Processing Systems Evaluation Workshop*, Report RLTR- 91-362, Rome Laboratory.

Jung, S. Y., Hong, S. L. & Paek, E. (2000). An English to Korean Transliteration Model of Extended Markov Window. *Proceedings of COLING 2000*.

KalyanaSundram, K. *Mylai Tamil Font for preparation of Texts in Tamil Script on Computers*, from website http://tamilelibrary.org/teli/mylai1.html.

Kang, B.J. & Choi, K.-S. (2000). Automatic Transliteration and Back-transliteration by Decision Tree Learning. *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, Athens, Greece.

Kanneganti, R. & Kishore, A. *Rice Inverse Transliterator (RIT) Software*, from website http://www.teluguworld.org/RIT/rit.html.

Knight, K. & Graehl, J. (1998). Machine Transliteration. *Computational Linguistics*, 24(4).

Lee, C. & Chang, J. S. (2003). Acquisition of English-Chinese Transliteration Word Pairs from Parallel-Aligned Texts using a Statistical Machine Translation Model. *Proceedings*

of HLT-NAACL Workshop: Building and Using parallel Texts Data Driven Machine Translation and Beyond, 2003, Edmonton, 96-103.

Malik, M.G.A. (2006). Punjabi Machine transliteration. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, 1137-1144.

Meng, H. M., Lo, W., Chen, B., & Tang, K. (2001). Generate Phonetic Cognates to Handle Name Entities in English-Chinese cross-language spoken document retrieval. *Proceedings of ASRU,* 2001.

Odell, M.K., & Russel, R.C. (1973). US Patent 1261167, 1918, cited in Knuth (1973) "Sorting and Searching: The art of computer programming" Volume 3, Addison Wesley, reading, MA.

Oh, J.H., & Choi, K.S. (2002). An English-Korean transliteration model using pronunciation and contextual rules. *Proceedings of the 19th international conference on Computational linguistics,* 1, 1-7.

Saini, T. S., & Lehal, G. S. (2008). Shahmukhi to Gurmukhi Transliteration System: A Corpus based Approach. *Research in Computing Science* (Mexico), 33, 151-162.

Srinivasan. (1995). "ADHAWIN", in "Transliteration schemes" from website http://acharya.iitm.ac.in/multi_sys/transli/schemes.php.

The Devanagari Script Block. (2008). from website http://people.w3.org/rishida/uniprop32/descn-devanagari.html accessed on 7 Feb 2008.

Transliteration Principles. (2008). from http://acharya.iitm.ac.in/multi_sys/translit.php accessed on 7 Feb 2008.

Virga, P., & Khudanpur, S. (2003). Transliteration of proper names in cross-language applications. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, 365-366.

Viswanadha, R. (2002). Transliteration of Tamil and Other Indic Scripts. *Tamil Internet 2002*, California, USA.

Wan, S., & Verspoor, C. M. (1998). Automatic English-Chinese name transliteration for development of multilingual resources. *Proceedings of COLING-ACL'98.*

Writing Systems. (2010). form website http://www.omniglot.com/writing/alphabets.htm.

Zhang, Y., & Vogel, S. (2004). Measuring Confidence Intervals for the Machine Translation Evaluation Metrics. In I*nternational Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2004)*. Baltimore, MD.