Mapping and Generation Model for Named Entity Transliteration in CLIR Systems

V. Narasimhulu, P. Sujatha and P. Dhavachelvan

Department of Computer Science, Pondicherry Central University,
Pondicherry - 605014, India.
{narasimhavasi, spothula, dhavachelvan}@gmail.com

Abstract: Named Entities are the expressions in human languages that explicitly link notations to the entities in the real world. They play an important role in Cross Lingual Information Retrieval (CLIR), because most of the user queries contain Out Of Vocabulary (OOV) terms with majority of named entities. Missing their translations has a significant impact on the retrieval system. Translation of named entities can be done using either named entity translation or named entity transliteration. Named entity translation causes translation failures, since if a given name is not found or new to the translation system, it may be discarded or mistranslation occurs. Transliteration is the suitable method for translating named entities and is a challenging task. This paper, discusses various transliteration techniques and a problem in the Compressed Word Format (CWF) mapping model. A system is proposed based on mapping and generation models. The proposed system follows one of the major techniques of transliteration called grapheme-based transliteration model. It transliterates all the source language names which are specified by the user and gives the right equivalent and relevant target names.

Keywords: Named entities, cross lingual information retrieval, translation, transliteration, grapheme.

1. Introduction

CLIR is the process of submitting query in source language and retrieving information in target language. This processing requires mainly three phases: Text processing, query translation and retrieval system. Text processing includes morphological analyzer, stop words removal and stemming process. Morphological analyzer analyzes the structure of the words in a given query. E.g. verbs, adverbs, adjectives etc. Stop words removal removes the stop words in a given query. E.g. the, is, was, can, should etc. Stemming is the process of reducing inflected words to their base or root form. E.g. fishing, fished and fisher are reduced into the root word fish. After the text processing, the source query contains a combination of dictionary and OOV words. Translation of dictionary words can be done using bilingual dictionary. Translation of OOV terms cannot be handled using bilingual dictionary because of its limited coverage. OOV words are significant for retrieving information in a CLIR system. The retrieval effectiveness can reduce up to 60% if OOV terms are not handled properly [1].

OOV terms can be of many types; some of them newly formed words, loan words, abbreviations or domain specific terms, but the biggest group of OOV terms, which are observed to be, as many as half of the whole observed OOV terms in [1], belongs to a group called named entities. Named entities are the expressions in human languages that explicitly link notations in languages to the entities in the real world. Examples of named entities are individual name, role, organization, location, date, time, quantity, numerical expression, phone number etc.

Named entities form a very dynamic set, already there exists a large quantity of them, and at the same time people are creating new named entities every day. This makes that dictionary cannot cover all named entities. They play important role in locating relevant documents. The occurrence of named entities in user queries makes easier for retrieval system, if the correct translations of named entities are available [2]. The retrieval performance or average precision of CLIR reduces significantly, when named entities are not translated properly in queries [3]. Translation of named entities can be done using named entity translator, which causes translation failures. i.e. either drops the word or mistranslates, if the given name is new to the translation system. Another possibility for translation of named entities is named entity transliteration. Previous studies [4] show that the average precision score of a CLIR system get reduced by 50% when the named entities were not properly transliterated. Therefore transliteration of named entities from source language to the target language presents an interesting challenge.

Transliteration is the process of transforming a word written in a source language into a word in a target language without the aid of a resource like a bilingual dictionary. It refers to expressing a word in one language using the orthography of another language. Orthography means the art or study of correct spelling according to established usage. Transliteration can be classified into two directions: forward transliteration and backward transliteration. Given a pair (s, t), where s is the source word in source language and t is the transliterated word in target language. Forward transliteration is the process of converting s into t. Backward or back transliteration is the process to correctly find or generate s for a given t. This paper is used in the work of forward transliteration.

The major techniques for transliteration can be classified into three categories: grapheme-based, phoneme-based and Hybrid transliteration models [5].

Grapheme refers to the basic unit of written language or smallest contrastive units. In grapheme-based transliteration spelling of the original string is considered as a basis for transliteration. It is 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. Here transliteration is identified by mapping the source language names to their equivalent names in a target language and generating them.

Phoneme refers to the simplest significant unit of sound or the smallest contrastive units of a spoken language. In phoneme-based transliteration pronunciation rather than spelling of the original string is considered as a basis for transliteration. Phoneme based transliteration is referred as a pivot method because it uses source language phonemes as a pivot, when it produces target language graphemes from source language graphemes. It usually needs two steps:

- Produce source language phonemes from source language graphemes.
- Produce target language graphemes from source phonemes.

These two steps are explicit if the transliteration system produces target language transliterations after producing the pronunciations of the source language words; they are implicit if the system uses phonemes implicitly in the transliteration stage and explicitly in the learning stage [6]. ARPAbet symbols are used to represent source phonemes. ARPAbet is one of the methods used for coding source phonemes into ASCII characters [7]. It is developed by Advanced Research Projects Agency (ARPA) as part of their Speech Understanding Project. ARPAbet symbols for English consonants are given in Table 1, and for English vowels are given in Table 2.

Table 1. Arpabet symbols for English consonants

P	T	K	B	D	G	M	N	NG	F
V	TH	DH	S	Z	SH	ZH	CH	JH	L
W	R	Y	H	Q	DX	NX	EL		

Table 2. Arpabet symbols for English consonants

IY	IH	EY	EH	AE	AA	AO	UH	OW	UW
AH	ER	AR	AW	OY	AX	AXR	IX	UX	

Grapheme-based and phoneme-based transliteration is referred to as hybrid transliteration. It makes use of both source language graphemes and phonemes, to produce target language transliterations. Here after, a source language grapheme is a source grapheme, a source language phoneme is a source phoneme, and a target language grapheme is a target grapheme.

In each model, transliteration of a source language to target language is an interesting and challenging task. The present paper discusses different issues and problems regarding transliteration from source language to target language and also problem in the CWF mapping model. The present paper proposes a system based on grapheme-based transliteration model.

The organization of the paper is as follows. In Section 2, the related works on different transliteration models and transliteration of named entities are described. Description of CWF mapping model, problem in the CWF mapping model and problem definition are given in the same Section. The description of the proposed system and comparison of proposed system with existing systems is given in the same Section 3. Section 4, describes about implementation details of the mapping and generation model. The conclusion of the paper is given in Section 4.

2. Related Work

Transliteration is the process of transcribing words from the source script to a target script. Grapheme-based transliteration considers the spelling or characters of the original string as the basis for transliteration. Previous studies have proposed several methods with this model. An n-gram based statistical transliteration model for English to Arabic names was described in [4]. It presents a simple statistical technique, which does not require any heuristics or linguistic knowledge of either language. It is specified that transliteration either of OOV named entities or of all OOV words is an effective approach for CLIR. A decision tree based transliteration model [8], is a language independent methodology for English to Korean transliteration and supports back transliteration. It is composed of character alignment and decision tree learning. Transliteration and back transliteration rules are induced for each English alphabet and each Korean alphabet. A maximum entropy based model [9] is an automatic transliteration model from English to Japanese words and it successfully transliterates an English word not registered in any bilingual or pronunciation dictionaries by converting each partial letters in the English word into Japanese characters. A new substring based transliteration method based on phrase-based models of machine translation was described in [10]. Substring based transliteration method is applied on Arabic to English words.

Phoneme-based transliteration considers the pronunciation of the word as the basis for transliteration. Previous studies have proposed several methods with this model. A rule based model for English to Korean transliteration using pronunciation and context rules is described in [11]. It uses phonetic information such as phoneme and its context as well as orthography of English language as the basis for transliteration. A machine-learned phonetic similarity model [12] is a backward transliteration model and provides learning algorithm to automatically acquire phonetic similarities from a corpus. Given a transliterated word, similarity based model compares the list of source candidate words and the one with highest similarity will be chosen as the original word. The first semantic transliteration of individual names was given in [13]. It is a phoneme-based transliteration and based on the word's original semantic attributes. It is a probabilistic model for transliterating person names in Latin script into Chinese script. Proved semantic transliteration substantially improves accuracy over phonetic transliteration.

Hybrid transliteration is a combination of both grapheme-based and phoneme-based transliteration. Oh and Choi [5] proposed a model for improving machine transliteration using an ensemble of three different transliteration models for English to Korean and English to Japanese languages. Three transliteration models are grapheme, phoneme and both. Bilac and Tanaka [14] proposed a new hybrid back transliteration system for Japanese, which contains segmentation, phoneme-based and grapheme-based transliteration modules. The system first finds the best segmentation of transliterated string and then obtains back transliteration using the combined information based on pronunciation and spelling. Hong et al. [15] proposed a hybrid approach to English-Korean name transliteration. It is based on phrase-base Statistical Machine Translation (SMT) model with enabled factored translation features. The system is combined with various transliteration methods including a Web-based n-best re-ranking, a dictionary-based method and a rule-based method.

In the Indian language context, transliteration similarity mechanism to align English-Hindi texts at the sentence and word level in parallel corpora was given by [16]. This is based on a grapheme-based model. It describes a simple sentence length approach to perform sentence alignment and multi feature approach to perform word alignment. Punjabi machine transliteration was given by [17]. It addresses the problem of transliteration for Punjabi language from Shahmukhi (Arabic script) to Gurmukhi using a set of transliteration rules (character mappings and dependency rules). A discriminative, Conditional Random Field (CRF)–Hidden Markov Model (HMM) model for transliterating words from Hindi to English was used in [18]. The model is a statistical transliteration model, which generates desired number of transliterations for a given source word. It is based on grapheme-based model and language independent. A word origin based transliteration for splitting Indian and foreign origin words based on their phoneme equivalents was shown by [19]. The given transliteration mechanism is applicable for Indian languages and shown that word origin is an important factor in achieving higher accuracy in transliteration. A phrase or grapheme-based statistical machine transliteration of named entities from English to Hindi using a small set of training and development data was shown by [20]. A CWF mapping model for transliterating named entities from English to Tamil was given by [21]. This is based on grapheme-based model in which transliteration equivalents are identified by mapping the source language names to their equivalents in target language database.

2.1 Named Entity Recognition (NER)

NER is the subtask of the information extraction that seeks to locate and classify atomic elements in a text into predefined categories. It extracts the specific information from the text or document. It has important significance in internet search engines and performs important tasks in many of the language engineering applications such as machine translation, Question-Answering (QA) systems, indexing for information retrieval and automatic summarization. Named entity recognizer is language dependent. Each language needs a separate named entity recognizer. The following shows a simple example for recognizing named entities in a text using named entity recognizer or tagger.

E.g. Ram bought 3000 shares.
<ENAMEX TYPE="PERSON">Ram</ENAMEX> bought <ENAMEX TYPE="QUANTITY">3000</ENMAEX> shares.

In the above example the annotations have been done using so called ENAMEX tags. A large number of techniques have been developed to recognize named entities for different languages. Some techniques are rule based and others are statistical or machine learning techniques [22], [24]. These techniques are summarized in Table 3.

Table 3. NER Techniques

S. No	Name of the Technique		Description
1	Rule Based Technique		It uses the morphological and contextual evidence of a natural language and a consequently determines the named entities.
2	Statistical Learning Techniques	Supervised Learning	In this NER process is learned automatically on large text corpora and the supervised by a human.
		Unsupervised Learning	In this NER process is not supervised, instead existing semantic lexical databases such as WordNet are consulted automatically
		Semi-supervised Learning	It involves a small degree of supervision, such as a set of seeds, for learning the process.

2.2 CWF Mapping Model

CWF mapping model [21] is a grapheme based transliteration model, where transliteration equivalents are identified by mapping the source language names to their target language database, instead of generating them. The basic principle is to compress the source word into its minimal form and align it across an indexed list of target language words to arrive at the top n-equivalents based on the edit distance. That is, for a given a source language named entity (English) string, it will produces a ranked list of transliterated names in the target language (Tamil).

In this model individual names in the target language are collected manually and listed in the database by running named entity recognizer on the archives. These names are then romanized so that they can be easily compared to the English queries. For a given English named entity, compress both English named entity and list of collected Tamil names into minimal consonant form based on a set of linguistic rules. Linguistic rules are an ordered set of rewrite and remove rules. Rewrite rules replace characters or clusters of characters with other characters or clusters. Remove rules simply remove the characters or clusters of characters. For mapping Tamil names, custom Romanization scheme was used. This scheme maps every Tamil character to Roman characters in a one-to-one mapping fashion. One-to-one fashion means each Tamil letter is considered as a single roman character. After compressing both source and list of target names in the database index, right equivalents were matched using the Modified Levenshtein algorithm [21], by calculating edit distance between source name and list of target names in the database index. Here mapping is done between compressed word of a source name and compressed word of a target name. Levensthtein edit distance is normally used for finding number of changes, between two strings of same language and contains insertion, deletion or substitution of a single character [23]. Modified Levenshtein is a variant of Levenshtein algorithm and modified based on considering character sets of source and target languages, since source and target language strings use different character sets. The edit distance between perfect matching pairs can be zero. The algorithm has been proved to be effective for matching perfect pairs. Instead of single perfect matching pair, CWF mapping model retrieve n-equivalent matching pairs based on the nearest edit distance value for CLIR applications.

2.2.1. Advantages of CWF mapping model

- CWF mapping model is more accurate than the statistical generation models in English-to-Tamil transliteration.
- In case of mapping based approaches, CWF is more accurate and precise than the actual word forms.
- Using CWF, half of the execution time is reduced. The reason is mapping is done between the compression words, not between the actual words.

2.3 Problem in CWF Mapping Model

CWF mapping model accurately map the right matching pair between source and target languages, only if a given source named entity has a right matching equivalent in the target language database index. Mapping is done between the compressed word of source name and compressed word of target name. Suppose a given source named entity does not have a right equivalent matching pair in the target language database, it gives n-relevant matching pairs based on the edit distance, not the required exact equivalent matching word, because, the previous model [21] would not be generating the target transliterations and not updating the database index regularly. Except collecting and listing the target language names in the database. The drawback of the system is: mapping is taking place only with the source names which are presented in the database. It would not work for other source names which are not in the database.

2.4 Problem Definition

To overcome the problem, a system is proposed based on mapping and generation. It is a grapheme-based transliteration model, which generates or transliterates the source name into equivalent target name, if the right equivalent is not available in database. After transliterating, web is used to retrieve relevant words for finding relevant equivalents and database is updated regularly

3. Proposed System

3.1 Introduction

A system is proposed for transliterating named entities from source to target language, which is based on major technique of transliteration called grapheme-based transliteration. The proposed system includes mapping and generation models. It transliterates given source name into equivalent target name and retrieves relevant words. This equivalent and relevant words are displayed to the user.

3.2 Proposed System Model

The overview of the proposed system is shown in Figure 1. It mainly composed with the following modules: Word Compression, Mapping, Transliteration, Web Retrieval and Updation. The working principle of individual modules can be described in the following paragraphs.

3.2.1 Word Compression Module

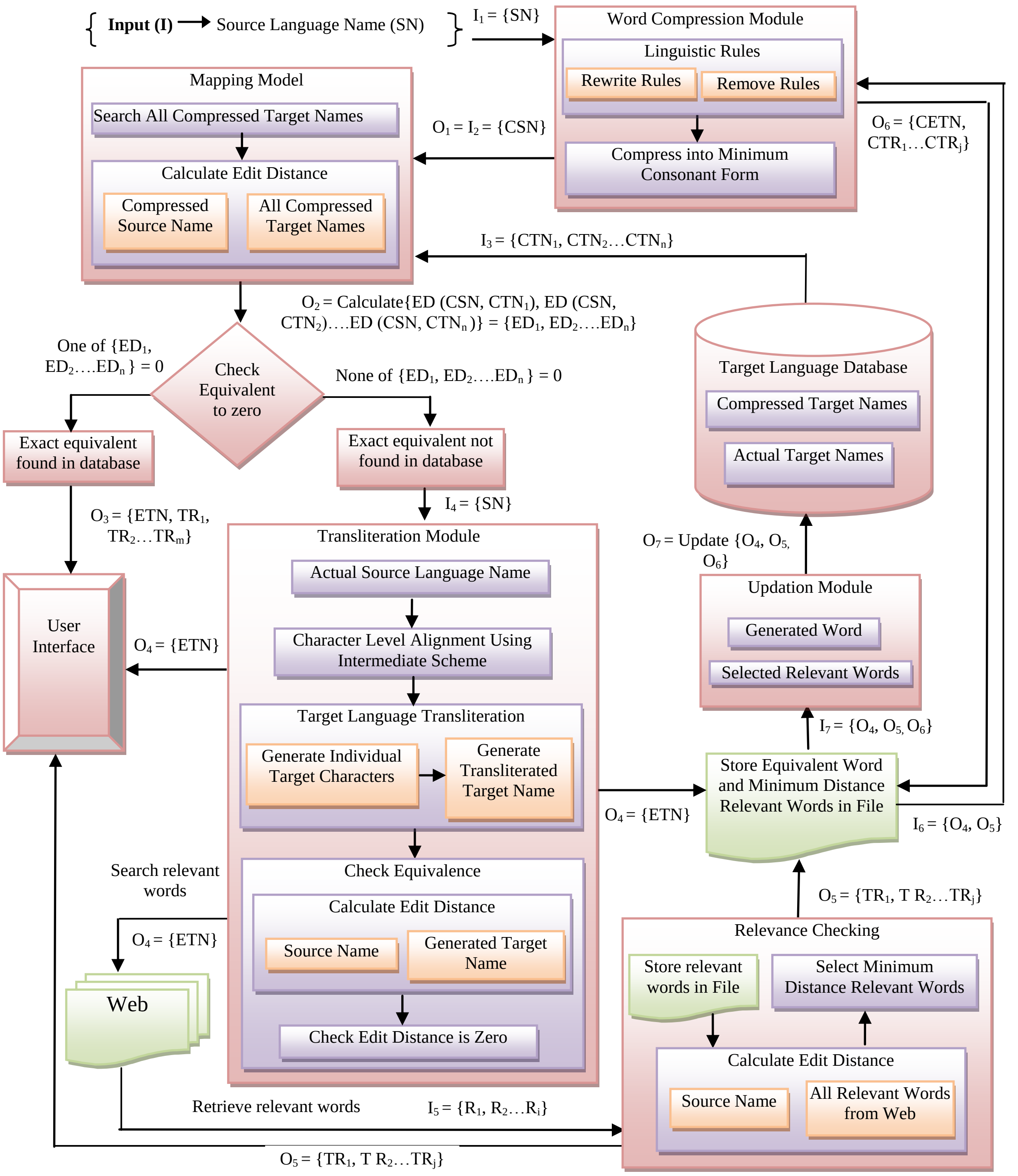
For a given source language name (SN), this module compress as it into a minimum consonant skeleton form or CWF form based on a set of linguistic rules. Linguistic rules are an ordered set of, combination of rewrite and remove rules as specified in [21]. The output of this module is named as compressed source language name (CSN). The target language names $\{TN_1, TN_2 \ldots TN_n\}$ in the database are compressed in the same manner by using the linguistic rules, but their rule sets are different. Target language database index is in the form of tuples, where each tuple contains both compressed name and actual name.

3.2.2 Mapping Model

This model receives CSN as input and searches all compressed target names $\{CTN_1, CTN_2 \ldots CTN_n\}$ from the database index. It converts CSN and $\{CTN_1, CTN_2 \ldots CTN_n\}$ into intermediate scheme for finding exact equivalent compressed target name (CETN). Intermediate scheme acts as a mediator for mapping between source and target languages because both have different character representations. Roman scheme is the intermediate scheme developed for mapping or aligning characters between source and target languages. A scheme having one-to-one configuration is used here for mapping between CSN and $\{CTN_1, CTN_2 \ldots CTN_n\}$. The model then calculates edit distance between compressed source name and each compressed target names $\{ED\ (CSN, CTN_1), (ED\ (CSN, CTN_2) \ldots (CSN, CTN_n)\}$.

Calculate$\{ED\ (CSN, CTN_1), (ED\ (CSN, CTN_2) \ldots (CSN, CTN_n)\} = \{ED_1, ED_2 \ldots ED_n\}$.

Modified Levenshtein algorithm is used for finding edit distances between CSN and $\{CSN_1, CSN_2 \ldots CSN_n\}$ as described in the paper [21]. After finding $\{ED_1, ED_2 \ldots ED_n\}$, each edit distance is checked with equivalent to zero for finding exact target equivalent. If one of $\{ED_1, ED_2 \ldots ED_n\}$ is equivalent to zero, then CETN is found. Therefore, the corresponding actual target name (ETN) and relevant actual target names $\{TR_1, TR_2 \ldots TR_m\}$ are retrieved from the database based on the minimum edit distance value. Finally ETN and $\{TR_1, TR_2 \ldots TR_m\}$ are displayed in the user interface. When there is more than one candidate at the same edit distance, finer ranking can be made based on the edit distance between the actual forms of source and target strings. There is no chance for two candidates having zero edit distance, because one string can have only one equivalent not more than one. Suppose none of $\{ED_1, ED_2 \ldots ED_n\}$ is equivalent to zero, exact ETN is not found


Input (I) → Source Language Name (SN)
I_1 = {SN}
Word Compression Module
Linguistic Rules
Rewrite Rules
Remove Rules
Compress into Minimum Consonant Form
$O_1 = I_2$ = {CSN}
O_6 = {CETN, $CTR_1 \ldots CTR_j$}
Mapping Model
Search All Compressed Target Names
Calculate Edit Distance
Compressed Source Name
All Compressed Target Names
I_3 = {CTN_1, $CTN_2 \ldots CTN_n$}
O_2 = Calculate{ED (CSN, CTN_1), ED (CSN, CTN_2)….ED (CSN, CTN_n)} = {ED_1, $ED_2 \ldots ED_n$}
Target Language Database
Compressed Target Names
Actual Target Names
One of {ED_1, $ED_2 \ldots ED_n$ } = 0
Check Equivalent to zero
None of {ED_1, $ED_2 \ldots ED_n$ } = 0
Exact equivalent found in database
Exact equivalent not found in database
I_4 = {SN}
O_3 = {ETN, TR_1, $TR_2 \ldots TR_m$}
Transliteration Module
Actual Source Language Name
Character Level Alignment Using Intermediate Scheme
Target Language Transliteration
Generate Individual Target Characters
Generate Transliterated Target Name
Check Equivalence
Calculate Edit Distance
Source Name
Generated Target Name
Check Edit Distance is Zero
O_7 = Update {O_4, O_5, O_6}
Updation Module
Generated Word
Selected Relevant Words
I_7 = {O_4, O_5, O_6}
User Interface
O_4 = {ETN}
Store Equivalent Word and Minimum Distance Relevant Words in File
O_4 = {ETN}
I_6 = {O_4, O_5}
O_5 = {TR_1, T $R_2 \ldots TR_j$}
Search relevant words
O_4 = {ETN}
Web
Relevance Checking
Store relevant words in File
Select Minimum Distance Relevant Words
Calculate Edit Distance
Source Name
All Relevant Words from Web
Retrieve relevant words
I_5 = {R_1, $R_2 \ldots R_i$}
O_5 = {TR_1, T $R_2 \ldots TR_j$}

I = Input Information Flow	O = Output Information Flow
I_1 = {SN} = {Source Language Name}	O_1 = {CSN} = {Compressed Source Name}
I_2 = {CSN} = {Compressed Source Name}	O_2 = Calculate{ED (CSN, CTN_1)….ED (CSN, CTN_n)} = {ED_1, $ED_2 \ldots ED_n$} = {Edit Distances between CSN and Compressed Target Names}
I_3 = {CTN_1, $CTN_2 \ldots CTN_n$} = {Compressed Target Names in database}	O_3 = {ETN_1, TR_1, $TR_2 \ldots TR_m$} = {Equivalent Target Name and Relevant Target Names}
I_4 = {SN} = {Source Language Name}	O_4 = {ETN} = {Equivalent Target Name}
I_5 = {R_1, $R_2 \ldots R_i$} = {Retrieved Relevant Words}	O_5 = {TR_1, T $R_2 \ldots$, TR_j} = {Minimum Distance Relevant Words}
I_6 = {O_4, O_5} = {ETN, TR_1, $TR_2 \ldots TR_j$} = {Equivalent Target Name and Minimum Distance Relevant Words}	O_6 = {CETN, CTR_1, $CTR_2 \ldots CTR_j$} = {Compressed both Equivalent Target Name and Minimum Distance Relevant Words}
I_7 = {O_4, O_5, O_6} = {ETN, TR_1, $TR_2 \ldots TR_j$, CETN, CTR_1, $CTR_2 \ldots CTR_j$}	O_7 = Update {O_4, O_5, O_6} = {Updating Equivalent and Minimum Distance Relevant Target words in the Actual and Compressed Form in Database}

Figure 1. Mapping and generation model for named entity transliteration

and there is a need to transliterate SN for ETN. Only, relevant words can be retrieved based on the minimum edit distance, but not the required ETN.

3.2.3 Transliteration Module

It generates or transliterates source language named entity into target language name which is not having ETN in the target language database index. Transliteration is done on the SN, not on the CSN. This work used the major technique of transliteration called grapheme based model, where transliterations are generated using character level alignments between source and target languages. Intermediate scheme is used to align the characters between

source and target languages. The roman scheme is used as intermediate scheme for transliteration. Mapping is done with either one-to-many and one-to-one configuration or many-to-one and one-to-one configuration, which is because of given source name is in SN not in the CSN. The selection of the configuration depends on source and target languages. The output of transliteration module is the generation of ETN, which is not found in the database index. After generating the ETN, the edit distance is calculated between SN and ETN for finding how far generated transliteration is right equivalent. Finally, the generated ETN is displayed to the user interface and stored in a file for compressing and updating the database.

3.2.4 Web Retrieval Module

It is mainly designed for searching the relevant target words $\{R_1, R_2 \ldots R_i\}$ from the web for the generated ETN. After retrieving $\{R_1, R_2 \ldots R_i\}$, they are stored in a file. Edit distances are calculated between SN and $\{R_1, R_2 \ldots R_i\}$ for relevance checking. The words which have less or minimum edit distance are taken as relevant to the source name. The words that have minimum distance $\{TR_1, TR_2 \ldots TR_j\}$ are displayed in the user interface and stored in a same file. These names $\{ETN, TR_1, TR_2 \ldots TR_j\}$ are given to the Word Compression Module to compress into CWF by using a set of linguistic rules. These compressed names $\{CETN, CTR_1, CTR_2 \ldots CTR_j\}$ are stored into the same file, which it has $\{ETN, TR_1, TR_2 \ldots TR_j\}$ as shown in Figure 1.

3.2.5 Updation Module

It is mainly designed for updating the database with the generated ETN and retrieved target names from the web which has good relevance $\{TR_1, TR_2 \ldots TR_j\}$, so that they need not transliterate and search again. This module updates $\{CETN, CTR_1, CTR_2 \ldots CTR_j\}$ and $\{ETN, TR_1, TR_2 \ldots TR_j\}$ in the form of tuples that contains compressed name with the actual name. The processing time of Transliteration Module and Web Retrieval Module has been reduced, when already specified source name is given again for transliteration.

The final result contains both right equivalent matching target name (ETN) and relevant equivalent target names, which will be shown in the user interface.

$$\text{Input:} \quad \left\{\begin{array}{l} \text{Input1} \longrightarrow \text{Source Language Name-1 } (SN_1) \\ \text{Input 2} \longrightarrow \text{Source Language Name-2 } (SN_2) \end{array}\right\}$$

Output1: ETN Found in database

$$\left\{\begin{array}{l} \text{Equivalent Target Name (ETN)} \\ \text{Target Relevant Name-1}(TR_1) \\ \quad\quad\quad\quad\vdots \\ \text{Target Relevant Name-m } (TR_m) \end{array}\right\}$$

Output2: ETN not found in database

$$\left\{\begin{array}{l} \text{Equivalent Target Name (ETN)} \\ \text{Target Relevant Name-1}(TR_1) \\ \quad\quad\quad\quad\vdots \\ \text{Target Relevant Name-j } (TR_j) \end{array}\right\}$$

3.3 Performance of the System

The performance of the system can be predicted based on execution time of Word Compression Module, Mapping Model, Transliteration Module and Web Retrieval Module. Indexing is used for searching in the database. Indexing is the fastest method for searching. One of the indexing method, i.e. clustered indexing is used here. Clusters are divided based on the starting letter of the actual target names. For a given source name, need to search in the corresponding cluster of the target database to find equivalent and relevant words.

This type of indexing can reduce the searching time significantly in the database, so that efficiency of the system is improved. The system transliterates effectively for a given source word, which is not in the database. The system is accurate for mapping between compressed source name and compressed target name. It also precisely retrieves exact equivalent and relevant target names.

3.4 Comparison with Existing Systems

The comparison of proposed system with CWF mapping model [21] is given in Table 4. The comparison of proposed system with CRF-HMM (generation) model [18] and CWF mapping model with respective to the characteristics is given in Table 5.

Proposed system supports all the source names, where as CWF mapping model supports only the source names which have equivalents in database. It is designed for CLIR system, where one transliteration is not sufficient, requires more than one transliteration. For a given source name, mapping and generation model specifies top (more than one) relevant equivalents based on edit distance. Transliteration is the important part of the proposed system. As the number of top equivalents increasing, transliteration accuracy also increased. Proposed system has more transliteration accuracy than CRF-HMM model on top ranked relevant equivalents. Target database is designed with clustered index method, which improves the search process of the mapping and generation model, so that the efficiency of the system has been improved.

Table 4. Comparison of proposed system with existing system

S. No	Existing System (CWF Mapping Model)	Proposed System
1	Transliteration is based on only database target names.	Transliteration is based on other than database target
2	Grapheme based mapping model.	Grapheme based mapping and generation model.
3	Specifies top-n equivalents with relevance ranking.	Specifies top-n equivalents with relevance ranking.
4	Database is constructed manually and need not updated regularly.	Database is updated with the generated target name and relevant words in the compressed and actual forms.
5	Precision is less, if the given source name does not have equivalent target name in the database.	Precision is good because it is transliterating and giving the equivalent target name, if it is not in the database.
6	No exact equivalency for other than database names	Exact equivalency is good for all the source names.

Precision is one of the performance measures for retrieval systems. It specifies the measure of exactness. Here, it is used to measure the exact equivalence with respective to the source name. CWF mapping model has more precision on only database names, where as proposed system probably have good precision on all the names other than database. CRF-HMM model does not reduce the edit distance between source and target names, where as proposed system reduce.

Table 5. Comparison of characteristics with CRF-HMM model and CWF mapping model

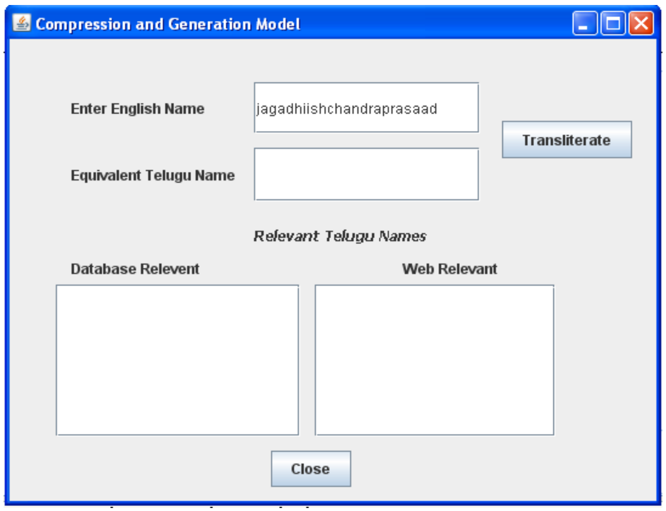
S. No	Characteristics	CRF-HMM Model	CWF Mapping Model	Proposed System
1	Generation model	Yes	No	Yes
2	Compression mapping model	No	Yes	Yes
3	Supports all the source names	Yes	No (Only database	Yes
4	Specifies top–n equivalents	Yes	Yes	Yes
5	Transliteration accuracy	Less	More (Only for database names)	More (For all the names)
6	Execution time	More	Less	Less for database names and more for other names
7	Exact Equivalence	Good (For all the names)	More (Only for database names)	Good (For all the names)
8	Reduce edit distance between source and target names	No	Yes (on compressed names)	Yes (on compressed names)

edit distance because of Modified Levenshtein algorithm and compressing the source and target names

4. Implementation and Evaluation

4.1 Implementation Details

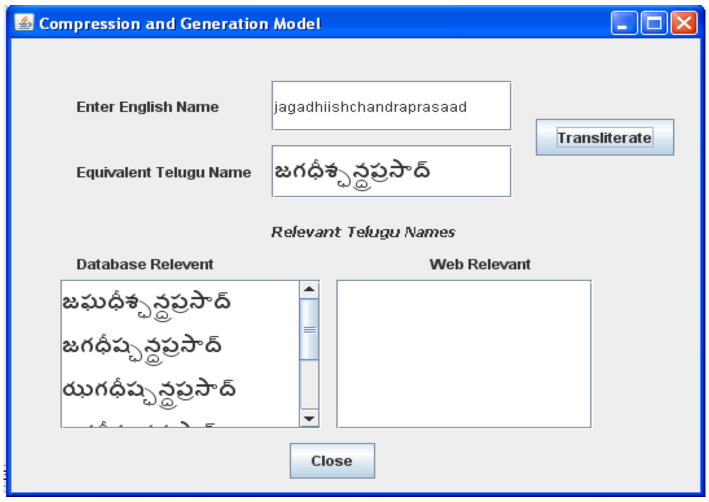
This section describes the implementation of the mapping and generation model. It is implemented using GUI (Graphical User Interface) components of the Java programming. MSAccess is used as a database for maintaining target language names. Unicode is used for storing the target names in the database. Unicode is a standard which provides a unique number for every character, no matter what the language is. Here, English language is used as a source language for receiving input from the user and Telugu language is used as a target language to display the output.



Enter English Name: jagadhiishchandraprasaad
Equivalent Telugu Name:

Figure 2. Source name specified by the user

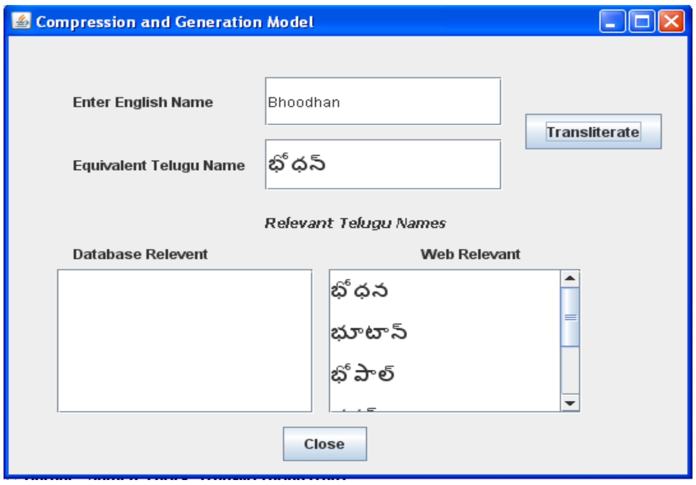
For a given English name as shown in Figure 2, mapping and generation model retrieves the exact equivalent and relevant Telugu names, if the given English name has exact equivalent Telugu name in the database and displays at the user interface as shown in Figure 3. The proposed model displays topmost 5 relevant Telugu names at the user interface and set the minimum edit distance is less than equal to two (<=2). Otherwise mapping and generation model transliterates the English name into equivalent Telugu name and retrieves the minimum edit relevant Telugu names (<=2) from the web and stored in a file. Finally, equivalent and relevant Telugu names are given to the user interface as shown in Figure 4, and they are updated in the database.



Enter English Name: jagadhiishchandraprasaad
Equivalent Telugu Name: జగధీశ్చన్ద్రప్రసాద్

Database Relevent:
జఘధీశ్చన్ద్రప్రసాద్
జగధీష్చన్ద్రప్రసాద్
ఝగధీష్చన్ద్రప్రసాద్

Figure 3. Equivalent and relevant target names from the database



Enter English Name: Bhoodhan

Equivalent Telugu Name: భోధన్

Database Relevent:

Web Relevant:
భోధన
భూటాన్
భోపాల్

Figure 4. Equivalent transliterated target name and relevant target names from the web

4.2 Evaluation Results

To evaluate performance and accuracy of the compression and generation model stored 800 Telugu named entities in the database. The list of different categories of named entities stored in the database is specified in Table 6.

Table 6. Categories of named entities stored

Category	Number of Entities
Person names	410
Place names	224
Organization names	166

Database is designed with both compressed and actual names of above category named entities. Totally 100 English named entities were used to verify the proposed system. In that, 50 named entities have equivalent Telugu names in the database. This is for evaluating the system whether it is correctly identifying the equivalent name and retrieving the relevant names. Remaining English named entities were used whether it is correctly transliterating into equivalent Telugu names and retrieving relevant names from the web. After above evaluation process, all the relevant Telugu names received from the web are updated in the database. From the above evaluation performance, equivalency and transliteration accuracy is calculated and given in Table 7.

Table 7. Evaluation of the proposed system

English Named Entities (100)	Performance	Equivalency	Transliteration Accuracy
Equivalent English named entities (50)	97.60%	98.74%	97.62%
Non-Equivalent English named entities (50)	95.96%	98.36%	97.04%

The above results clearly shows that the compression and generation model can be used for both equivalent and non equivalent named entities in the database.

5. Conclusion and Future Work

Named Entities are the expressions in human languages that explicitly link notations to the entities in the real world. They play an important role in CLIR. Transliteration is the relevant technique for translating named entities between source and target languages. All the major techniques and general problems in the transliteration are described in this paper. A system is proposed based on grapheme-based transliteration, which includes mapping and generation. It reduces the search processing time in the database because of clustered index. It improves transliteration accuracy and equivalency of the system by generating the target name and retrieving relevant words from the web effectively and efficiently.

Our current work can be extended further by integrating proposed model with the CLIR system and reducing the search process in the database index. An interesting future scope is going to use effective string matching methods for matching between source and target names. Finally, this work can be extended on more than one target language. That is Multi Lingual Information Retrieval (MLIR).

References

[1] D. Demner-Fushman and D. W. Oard, “The effect of bilingual term list size on dictionary-based cross-language information retrieval,” *In 36th Annual Hawaii International Conference on System Sciences (HICSS'03)*, pp. 108-118, 2003.

[2] T. Mandl and C. Womser-Hacker, “The effect of named entities on effectiveness in cross-language information retrieval evaluation,” *In ACM SAC'05*, pp. 1059-1064, 2005.

[3] L. S. Larkey, N. A. Jaleel, and M. Connell, “What’s in a Name?: Proper names in arabic cross language information retrieval,” *CIIR Technical Report*, IR-278, 2003.

[4] N. A. Jaleel and L. S. Larkey, “Statistical transliteration for english-arabic cross language information retrieval,” *In Proceedings of the twelfth international conference on Information and knowledge management*, November 03-08, 2003, New Orleans, LA, USA.

[5] J. H. Oh and K. S. Choi, “An ensemble of transliteration models for information retrieval,” *Information Processing and Management: an International Journal*, v.42 n.4, pp. 980-1002, July 2006.

[6] S. Bilac and H. Tanaka, “Improving back-transliteration by combining information sources,” *In Proceedings of IJC-NLP*, pp. 542–547, 2003.

[7] D. Jurafsky and J. H. Martin, “Speech and Language processing: An introduction to natural language processing,” *Computational Linguistics and Speech Recognition*, 2007.

[8] B. J. Kang and K. S. Choi, “Automatic transliteration and back- transliteration by decision tree learning,” *In:*

Proc. Of the Second International Conferenceon Language Resources and Evaluation, 2000.

[9] I. Goto, N. Kato, N. Uratani, and T. Ehara, “Transliteration considering context Information based on the maximum entropy method,” *In Proceedings Of the IXth Machine Translation Summit*, 2003.

[10]T. Sherif and G. Kondrak,”Substring based transliteration,” *In proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 944-951, 2007.

[11] J. H. Oh and K. S. Choi, “An English-Korean transliteration model using pronunciation and contextual rules,” *In: Proc. Of the 19th International Conference on ComputationalLinguistics*, pp. 758–764, 2002.

[12] W. H. Lin and H. H. Chen, “Backward machine transliteration by learning phonetic similarity,” *In: Proc. Of the Sixth Conference on Natural Language Learning*, pp. 139–145, 2002.

[13] H. Li, K. C. Sim, J.S. Kuo, and M.Dong, “Semantic transliteration of person names. *In proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 720-727, 2007.

[14] S. Bilac and H. Tanaka, “A hybrid back-transliteration system for Japanese,” *In: Proc. Of the 20th International Conference on Computational Linguistics* (COLING 2004), pp. 597–603, 2004.

[15] G. Hong, M. J. Kim, D. G. Lee, and H. C. Rim, “A Hybrid approach to English-Korean name transliteration,” *In proceedings of the Named Entities Workshop, ACL–IJCNLP’09*, pp. 108-111, August 2009.

[16] N. Aswaniand and R. Gaizauskas, “A hybrid approach to align sentences and words in English-Hindi parallel corpora,” *In Proceedings of the ACL Workshop on Building and Exploiting Parallel Texts*, 2005.

[17] M. G. A. Malik. “Punjabi machine transliteration,” *In Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pp. 1137–1144, 2006.

[18] S. ganesh, S. Harsha, P. Prasad, and V. Varma, “Statistical transliteration for cross language information retrieval using HMM aligment and CRF,” *In Proceedings of International Joint Conference on Natural Language Processing(IJCNLP)*, 2008, NERSSEAL Workshop, Hyderabad, India.

[19] H. Surana and A. K. Singh, ” A more discerning and adaptable multilingual transliteration mechanism for Indian languages,” *In Proceedings of International Joint Conference on Natural Language Processing (IJCNLP), 2008*, Hyderabad, India.

[20] T. Rama and K. Gali, “Modeling machine transliteration as a phrase based stastistical machine translation problem,” *In proceedings of the Named Entities Workshop, ACL–IJCNLP’09*, pp. 124-127, August 2009.

[21] S. C. Janarthanam, S. Sethuramalingam and U. Nallasamy, “Named entity transliteration for cross-language information retrieval using compressed word format mapping algorithm,” *In Proceedings of 2nd International ACM Workshop on Improving Non-English Web Searching (iNEWS08)*, CIKM-2008.

[22] A. Nayan , B. R. K. Rao, P. Singh, S. Sanyal, and R. Sanyal, “Named entity recognition for Indian languages,” *In Proceedings of International Joint Conference on Natural Language Processing (IJCNLP)*, pp. 97-104, 2008.

[23] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions and reversals,” *Sov. Phys. Dokl.*, vol. 6, pp. 707-710, 1966.

[24]F. Gralinski, k. Jassem, and M. Marcinczuk, “An environment for named entity recognition and translation ,” *In Proceedings of the 13th Annual Conference of the EAMT*, pp. 89-95, 2009.

Appendix

1. Psuedo code for Leveinshtein edit distance

Input: Two strings, X and Y
Output: The minimum edit distance between X and Y

```
M ← length (X)
N ← length (Y)

for i = 0 to m do
dist [i, 0] ← i

for j = 0 to n do
dist [0, j] ← j

for i = 0 to m do
for j = 0 to n do

dist [i, j] = min{ dist [i-1, j] + insert_cost ,
                   dist [i-1, j-1] + substitution_cost [ $X_i$, $Y_j$],
                   dist [i, j-1] + deletion cost
                 }
End
```

2. Romanization scheme for mapping Roman characters and Telugu (Contains one to one configuaration)

Vowels		Consonants					
a	అ	k	క	N	ణ	l	ల
A	ఆ	K	ఖ	t	త	L	ళ
i	ఇ	g	గ	T	థ	v	వ
I	ఈ	G	ఘ	d	ద	S	శ
u	ఉ	f	ఙ	D	ధ	R	ష
U	ఊ	c	చ	n	న	s	స
q	ఋ	C	ఛ	p	ప	h	హ
Q	ౠ	j	జ	P	ఫ		
V	ఎ	J	ఝ	b	బ		
e	ఏ	F	ఞ	B	భ		
E	ఐ	w	ట	m	మ		
o	ఒ	W	ఠ	y	య		
O	ఓ	x	డ	r	ర		
z	ఔ	X	ఢ	Y	ఱ		

3. Romanization scheme for transliteration between Roman characters and Telugu (Contains one to one and many to one configuration)

Vowels	Consonants		
A అ	KA క	NA ణ	LA ల
AA ఆ	KHA ఖ	TA త	LLA ళ
I ఇ	GA గ	THA థ	VA వ
II ఈ	GHA ఘ	DA ద	SHA శ
U ఉ	NGA ఙ	DHA ధ	SSA ష
UU ఊ	CA చ	NA న	SA స
R ఋ	CHA ఛ	PA ప	HA హ
RR ౠ	JA జ	PHA ఫ	
E ఎ	JHA ఝ	BA బ	
EE ఏ	NYA ఞ	BHA భ	
AI ఐ	TTA ట	MA మ	
O ఒ	TTHA ఠ	YA య	
OO ఓ	DDA డ	RA ర	
AU ఔ	DDHA ఢ	RRA ఱ	