Sādhanā (2018) 43:93
https://doi.org/10.1007/s12046-018-0828-8

Machine transliteration and transliterated text retrieval: a survey

DINESH KUMAR PRABHAKAR[1,*] and SUKOMAL PAL[2]

[1]Department of Computer Science and Engineering, Indian Institute of Technology (Indian School of Mines), Dhanbad 826004, India
[2]Department of Computer Science and Engineering, Indian Institute of Technology (Banaras Hindu University), Varanasi 221005, India
e-mail: dinesh.nitr@gmail.com

MS received 29 October 2016; revised 17 June 2017; accepted 10 September 2017; published online 7 June 2018

Abstract. Users of the WWW across the globe are increasing rapidly. According to Internet live stats there are more than 3 billion Internet users worldwide today and the number of non-English native speakers is quite high there. A large proportion of these non-English speakers access the Internet in their native languages but use the Roman script to express themselves through various communication channels like messages and posts. With the advent of Web 2.0, user-generated content is increasing on the Web at a very rapid rate. A substantial proportion of this content is transliterated data. To leverage this huge information repository, there is a matching effort to process transliterated text. In this article, we survey the recent body of work in the field of transliteration. We start with a definition and discussion of the different types of transliteration followed by various deterministic and non-deterministic approaches used to tackle transliteration-related issues in machine translation and information retrieval. Finally, we study the performance of those techniques and present a comparative analysis of them.

Keywords. Transliteration; informal information; natural language processing (NLP); information retrieval.

1. Introduction

Information retrieval (IR) is a field that helps users find useful information from large text collections. The field has become more important after the development of the World Wide Web (WWW) as the amount of on-line information is growing rapidly, making IR more challenging [1]. This helps people to find information in increasingly diverse settings. People nowadays not only passively search for information from the informative scholarly resources from the Web but also actively create, share, tag the multifaceted content, sometimes formally, but more often informally on different social media like Facebook, Twitter, Orkut, Google+ and so on [2]. The textual content in these social media is different from the traditional formal text content in size, format and character. It contains lots of personal babbling, slang, the use of numerals (even within textual words in SMS texts) and a mixture of different scripts (English and non-English), different languages even using a single script and so on and so forth. Moreover, such informal text is often short, personalized, cryptic, localized, temporal, opinionated and often biased (based on sex, religion, community, politics and finance-related factors) [2, 3].

Nowadays, online social networking has become a major medium for communication. A sizeable proportion of its users communicate in their regional languages but using script of a foreign language. For example, a lot of Indians prefer Roman script during writing on social sites. Several socio-cultural and technical reasons can be attributed for this, like the following:

- users of these devices are mostly educated, and are well-conversant in English (especially in Indian sub-continent, the medium of higher education is pre-dominantly English);
- keyboards available with computers, laptops and smart-phones in South Asian markets are by default English 'QWERTY' ones; online users are used with this keyboard;
- limited availability of hardware (keyboard) with native language support; even if it is available, people are not used to it and therefore use is less popular;
- different transliterations and/or keyboard softwares either do not render native scripts properly or are not freely available;

http://www.internetlivestats.com/.

*For correspondence

- the available native-language hardware and software tools are not sufficiently user-friendly and therefore pose serious difficulty to inexperienced users, etc.

The text written in a native language, but using a non-native script, mostly does not follow any standard spelling rule, but uses the orthography of the script based on pronunciation of the words. This process of phonetically transforming the words of a language into a foreign or non-native script is called transliteration. Transliteration, especially to the Roman script, is used more frequently on the Web not only for documents, but also for user queries that intend to search for these documents. These data require some pre-processing (translation and/or transliteration) before other natural language processing (NLP) techniques can be used. Transliteration is used mainly in machine translation (MT) and cross-lingual information retrieval (CLIR). Finch *et al* [4] conducted a large-scale real-world evaluation of the application of automatic transliteration in an MT system and demonstrated that using a transliteration system can improve MT quality when translating unknown words. This finding is also corroborated by others like Zhao *et al* [5] and El-Kahky *et al* [6].

Quite a number of transliteration mechanisms have been proposed for some non-English European languages, Russian [6–8] and East Asian languages like Chinese [9–12], Japanese [13–17], Korean [18–22], West Asian languages like Arabic [23–25] and the Persian [26, 27]. There have been some recent attempts on some Indian languages like Hindi [8, 28–39], Bengali [33, 40–42], Punjabi [43], Telugu [44], Kannada [29, 45, 46] and Tamil [29, 31, 47]. However, the present state-of-the-art of transliteration for Indian and other South Asian languages can be considered to be in the initial stage.

Transliteration simply converts a text from one script to another. It is not concerned about faithfully representing the sounds of the original; rather, it focusses on representing the characters with as much accuracy and unambiguity as possible.[1] As a technique, transliteration can be seen in two ways. When one writes native terms using a non-native or foreign script, it is called *forward transliteration*. For example, 'गुलाब' (in Devanagari script) is a Hindi word meaning *rose* in English. It can be transliterated (written) in Roman script as *gulab* , *gulaab* , *goolab* or in some other form. On the other hand, when one represents conversion of a term back to its native script from a non-native script, it is called *back-transliteration*. For example, *gulab* written in Roman script is back-transliterated to 'गुलाब' in its native script. Forward transliteration allows for creativity of the transliterator, whereas back-transliteration is ideally strict and expects the same initial word to be generated (with some exceptions, especially for East Asian languages). Although Karimi *et al* [48] give a good account of pioneering survey on machine transliteration, an enormous body of work has been done in this area in the recent past, especially after 2009, which is not considered there. Another survey by Antony and Soman [49] focussed on some of the early works on transliteration involving Indian languages.

In this survey, we attempt to emphasize the recent works in the realm of transliteration. This is particularly important as everyday huge amount of text data is being generated on the Web with multi-lingual content in the transliterated domain. People are increasingly expressing themselves in the social media, often using a mixture of different languages with ever-evolving vocabularies. The task of transliteration has, therefore, been more expanded and thus more challenging than before.

The rest of the article is organized in the following way. In section 2, we briefly outline the scope of this work. We discuss some basic concepts of transliteration in section 3. We describe various transliteration approaches in section 4. Section 5 discusses the recent research initiatives in the field of transliterated search and retrieval. Section 6 highlights evaluation metrics used in the transliteration domain in general. In section 7, we do a comparative study of different models for transliteration and retrieval. Section 8 is focussed on the status of transliteration research across languages. Section 8 concludes with directions for future work.

2. Scope

The transliteration is an increasingly popular phenomenon worldwide with the introduction of Web 2.0 and mobile devices. In different social media, people create, share, tag and search multifaceted data multi-lingually but mostly using the Roman script [2]. Even if we concentrate only on the textual data, huge amount of text is being generated on the Web, substantial portion of which is in transliterated domain. Although these texts are mostly informal, they do contain a good amount of information and therefore need to be studied. It has wide ramifications in low-resource languages in general, where Web presence is limited, specifically for Indian languages.

The user-generated texts can be in either pure *transliterated* text or in *code-mixed/code-switching* text. In the first case, terms in the sentences are from single language and written in non-native script, whereas in the second, candidate terms are from different languages and might be in more than one language. Although transliterated text processing involves a pair of languages, code-mixed text may need more than two languages. Information access in this case is even more complex as language identification is a challenge followed by transliteration.

Following are some applications areas of the transliteration.

- **MT:** Transliteration has traditionally been used in MT for transforming the named entities (NEs) to the target

[1] https://en.wikipedia.org/wiki/Transliteration.

script by preserving their phonetic behaviour where target script may or may not be the native. During translation, NEs are identified from the sentences and to keep their phonetic aspects intact in target languages, they are transliterated. MT is also useful in CLIR.

- **Mixed-script information retrieval (MSIR):** Often the text contains multiple scripts involving multiple languages. In the trivial case, each language may use its own native script within a single document. Language identification can be done from the script itself and language-specific processing can be done. If there is purely transliterated text (monolingual using non-native script), the script is converted to the native one keeping the phonetic aspects intact. However, as there is no standard for spelling in the transliterated domain, spelling variations are a major challenge. IR systems encounter term mis-matching issues between queries and documents. Spelling variations can occur across queries and documents, even within a single document. To resolve them and bringing them to a common form is an important research problem.
- **Code-mixed information retrieval (CMIR):** Sometimes, two or more languages are present but not necessarily in their native scripts. Even within a single sentence there can be two or more scripts using a single or more languages and there is not necessarily native language–script mapping. Information search in such code-mixed domain faces multilingual issues and term mis-matching. Since the queries and/or documents may come from different languages the identification of language is important to transliterate/translate that term to native script/language. This combined approach can help address the issues of CMIR.

However, scope of the transliteration is not restricted to the domains listed earlier but many other areas of information processing. The study is more important for development of different linguistic tools in the low-resource languages, specifically in the Indian context. India is a multi-lingual country having several hundreds of languages. Most educated people know and use more than one language with English being the *lingua franca* in their communication. In this survey we have covered the transliteration approaches for the Hindi, Marathi, Bengali, Telugu, Kannada and Urdu along with the approaches for the foreign languages including Japanese, Chinese, Korean, Russian, Turkish and Hebrew. This paper may be helpful for the researchers working on a broad spectrum of NLP including IR/IE.

3. Basic concepts

Before going into the details of transliteration techniques, let us discuss some basic concepts and terminologies related to the domain.

Phoneme: Phonetics is the study of human speech. The phonetic representation of a sound is represented using []. A *phoneme* is the smallest unit of speech that changes the meaning of a word and it can be represented within / /. For example, if we substitute the sound [m] with [c] in the word ‘mock’ [mock], the word changes to ‘cock’ where /m/ is a phoneme.

Grapheme: A *grapheme* is the elementary unit of written language such as alphabetic letters, numerals, punctuation marks and symbols. In a phonemic orthography, a grapheme corresponds to one phoneme. In spelling systems that are non-phonemic (such as the spellings used most widely in written English), multiple graphemes may represent a single phoneme. Phonemes are called digraphs when there are two graphemes for a single phoneme and tri-graphs when there are three graphemes and so on. For example, the word ‘fish’ contains four graphemes (f, i, s and h) but only three phonemes, because ‘sh’ is a digraph.

Syllable: A *syllable* is a unit of pronunciation. It is formed with a syllable peak, which is often a vowel, with optional initial and final margins, which are mostly consonants. A word that consists of a single syllable (e.g. dog) is called a monosyllabic. Similarly, terms that include two syllables (disyllable) are called di-syllabic; for three syllables (trisyllable), they are called tri-syllabic. In general, polysyllables (and polysyllabic) may refer either to a word of more than three syllables or to any word of more than one syllable [48].

Writing system: A *writing system* is used to represent expressible elements or statements in languages. Any writing system has some specifications: a set of defined symbols called characters or graphemes, and a set of rules and conventions that assign meaning to the graphemes. There are five distinct writing systems based on functional classification: logo-graphic, syllabic, featural, alphabetic or segmental and ambiguous.

Logo-graphic writing system uses logo-grams, where a single written character is used to represent a complete grammatical word. Most Chinese characters are logo-grams.

Syllabic writing system defines a syllabary as a set of written symbols that represent exactly or approximately syllables that constitute words. Symbols in a syllabary typically represent either a consonant sound followed by a vowel sound, or a single vowel. Japanese writing system falls into this category.

Featural writing systems contain symbols that do not represent whole phonemes, but rather the elements or features that collectively constitute the phonemes. Only Korean Hangul is a featural writing system. Hangul has three levels of phonological representation: featural symbols, alphabetic letters (combined features) and syllabic blocks (combined letters).

Alphabetic or *segmental writing systems* contain alphabets that are a small set of letters or symbols that represent a phoneme of a spoken language. The Brahmic family and its derivatives, Arabic and Latin writing systems, are segmental.

Ambiguous writing systems are not purely of one type; instead, they use symbols of different writing systems. The writing system of most languages falls into only one of the previous categories. However, there are some languages that use more than one writing system (such as the English system). For example the English system includes numerals and other logo-grams such as $, & and #.

3.1 *Challenges in machine transliteration*

Karimi *et al* [48] listed some common challenges that machine transliteration systems encounter, in general. They can be classified into five categories: script specifications, missing sounds, transliteration variants, language of origin and deciding on whether or not to translate or transliterate a name (or part of it). These categories are described later in brief. For detailed discussion, readers are suggested to refer section 3 in [48].

3.1a *Script specifications:* A script is a representation of one or more writing systems, and is composed of symbols used to represent a text having a common characteristic. One script can be be used for several languages (e.g., Latin script is used for languages of Western Europe; Arabic script for Arabic, Persian, Urdu, Pashto, Malay and Balti: Devanagari for North Indian languages, etc.). On the other hand, some written languages require multiple scripts (Japanese is written in the Hiragana, Katakana syllabaries and the Kanji ideographs). To computationally process such different language scripts, one requires the knowledge of different character encodings for the symbols.

Another aspect of language script is direction of writing: left-to-right (LTR) or right-to-left (RTL). For example, Arabic, Persian, Hebrew and Taana scripts follow RTL, whereas the languages using Devanagari or Roman alphabets follow LTR. Transliteration systems that manipulate characters of the words should carefully handle two different directions.

3.1b *Missing sounds:* Each language has its own sound structure, and symbols of the language script map these sounds. If there is a missing sound in the letters of a language, single sounds are represented using digraphs and trigraphs. For example, the fresh digraph 'sh' corresponds to the sound [ʃ], where 'h' does not have any sound. Transliteration systems are expected to learn (usually in their training step) both the convention of writing the missing sounds in both source and target languages involved, and the convention of exporting the sounds from one language to the other.

3.1c *Transliteration variants:* Transliteration allows several variants of a source term to be valid, based on the opinions of different human transliterators who may have different dialects. For example, 'गुलाब' can be transliterated as *gulaab* or *goolaab* or *gulab* and so on. Obtaining all possible variants for all of the words in one corpus is not feasible because of the following reasons.

- Not all speakers of those languages can be called upon in the evaluation process.
- There is no particular standard for such a comparison, other than conventions developed among notions.

Further, introduction of new names of companies, products and people makes getting any standard transliteration difficult. Therefore, evaluation of transliteration systems becomes problematic, in particular when comparing the performance of different systems.

3.1d *Transliterate or not:* Deciding on whether transliteration is required for a name (or part of it) is a challenge for MT systems. NEs are out-of-dictionary words where both translation and transliteration can be necessary. For example, when a NE such as 'Congress Parliamentary Committee' is encountered in a text, the first word 'Congress' needs transliteration and next two words should be translated, which indicates partial transliteration. Another example may be with multi-word names; part of them, may be a word with meaning that should not be translated (for example, Amitabh Bachchan). We are bound to transliterate all the words of this name. Also, there is no capitalization for NEs in some languages (almost all Asian languages, Arabic and Persian) unlike English. Some studies [24, 50, 51] discussed this problem specifically.

4. Transliteration approaches

There are two ways of obtaining transliteration: by generation and by mining. Transliteration generation is the process of automatically generating transliterations for a given term (word) in a language script (say, A) into its counterpart in another language script (say B). This transformation is based on language-dependent mapping rules between scripts A and B according to pronunciation. On the other hand, transliteration mining is the process of extracting (mining) transliteration pairs from different resources, either parallel or comparable corpora, or the Web between A and B.

Substantial work has been done on both the techniques. Apart from these two, there are some approaches that combine both generation and mining for transliteration. We call them fusion approaches, although in some of the papers they are referred to as hybrid approaches. The hierarchical diagram (figure 1) shows different approaches. In this

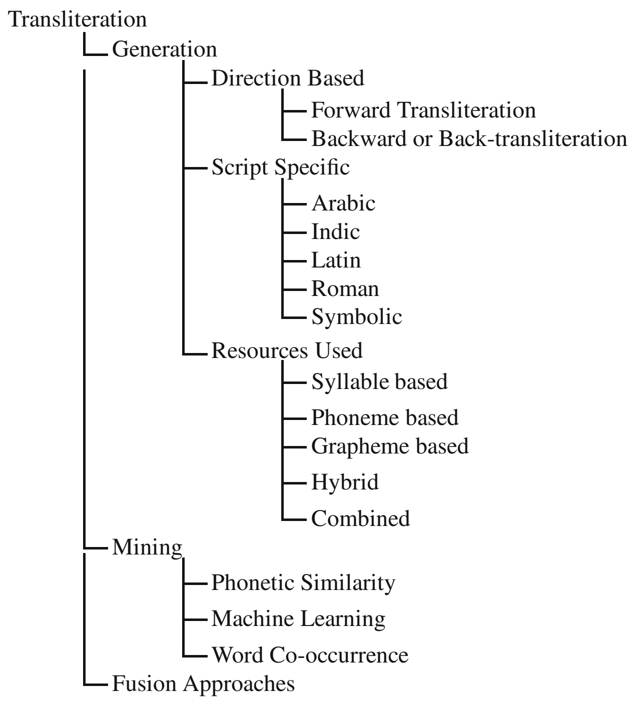

Figure 1. Classification summary of transliteration approaches.

section we discuss different approaches falling in each such category.

4.1 *Transliteration generation*

A number of generative transliteration methods have been proposed in the literature. Due to various attributes, such as the direction of transliteration, scripts of different languages and different information sources used, further subcategorization can be done, which is shown in figure 1 and described later with strengths and weaknesses of the schemes.

4.1a *Direction-based transliteration:* Based on direction, transliteration is of two types: forward and backward transliteration as discussed before.

4.1b *Script specific:* Generative transliteration approaches can also be categorized based on the script of languages for which they are proposed, e.g., languages with Latin script, languages with symbolic scripts, languages with Arabic script and languages with Indic scripts (such as Devanagari, Telugu and Bengali script for some Indian languages). Most research for languages with similar scripts is devoted to cross-lingual spelling variants, and their application in search tasks. Transliteration between languages that are widely different in script is more challenging as there are missing sounds and loss of information due to non-compatible phoneme equivalents [52].

4.1c *Based on resources used:* Generative transliteration approaches can also be divided based on the information sources used in the process into the following:

- *phonetic*-based approaches consider the task as a purely phonetical process and therefore use phonetics;
- *spelling*-based approaches consider it as an orthographic process and use grapheme handling techniques;
- *hybrid* approach (combination of both phonetic- and spelling-based approaches) and
- *combined* approaches (combination of any number of the spelling- or phonetic-based but not both.

4.1d *Survey on generation techniques:* There have been a number of works using transliteration generation. Transliteration models have been proposed for the transformation mainly between English and non-English languages, including Arabic [23, 53–55], Persian [26], Korean [20, 21, 56, 57], Chinese [11, 58–63], Japanese [15, 64–68] and Roman languages [69–72], which are discussed in the seminal survey by Karimi *et al* [48]. Here we will mainly discuss recent papers but include a few older ones that are not considered in the earlier survey.

Early work

One of the earliest papers in machine transliteration was by Knight and Graehl [73]. The process goes like this: initially a phrase is written in English, pronounced in English by the translator, its sound phonetic is adjusted to fit as per Japanese inventory and then sounds are transformed into Katakana. In this seminal work, the authors devised a generative model for back-transliteration of OCR'd names and technical terms in Japanese Katakana into actual English counterparts. The authors implemented modular learning approach for five different probability distributions using weighted finite state acceptor (WFSA) and weighted finite state transducers.

Kawtrakul *et al* [52] proposed a back-transliteration model from Thai to English. Thai documents, especially science and technical documents, contain words borrowed from other languages. Since the Thai script does not have any marker to identify foreign words, and does not contain space between words, distinguishing them from native words is difficult. The authors broke the words into syllables, which were matched with a Thai dictionary. Non-matching syllables were identified as coming from borrowed words. These syllables were then mapped to phonemes according to a set of hand-crafted transcription rules. Finally, English words were obtained from an English dictionary using fuzzy matching.

Syllable based

Jung *et al* [20] presented a statistical model for English to Korean transliteration generation. The model generates transliteration candidates probabilistically based on a set of rules. The rules were framed on the basis of pronunciation of alphabets, and previous, current and next context window. The model was designed based on conventional Markov windows and used a syllabic approach for English to Korean term alignment. The empirical accuracy was 93.90% on training data and 87.50% on test data, where the best 10 candidates were considered [20]. The pronunciation corpora is used for the existing terms and a transcription automata is used for the out-of-vocabulary (OOV) terms. Though it generates transliteration for the OOV terms correctly, it suffers from some limitations. The model fails to generate the transliteration when terms do not fit for the hand-crafted rules considered.

Phoneme based

Oh and Choi [21] proposed a pronunciation and contextual rule-based English to Korean transliteration model. The system operated in two phases: first, alignment of English Pronunciation Units (EPUs) with their corresponding phonemes and second, transliteration using some steps. With the help of a pronunciation dictionary, pronunciation units were identified for a given English word (for words in the dictionary) and then aligned with corresponding phoneme. Words not in the dictionary (assumed to be compound words) were split into two words and searched for in the dictionary followed by word alignment. After splitting, if the words could not be found, it was then checked whether they were English words of Greek origin. In the second phase, transliterations were generated using English-to-Korean Standard Conversion Rules (EKSCR). The method gained about 16% over the baseline in word accuracy (precision) [21]. The model emphasizes on phonetic, word formation and orthographic information. Using all the three together increases the number of transliterations since phoneme keeps sound information associated with individual words. Although it bettered state-of-the-art performance scores, the model failed to address some ambiguous phonemes encountered, especially in vowel sounds, e.g., 'AH'.

Virga and Khudanpur [12] proposed an English-to-Chinese NE transliteration model for CLIR. The task was accomplished in four steps. First, conversion of the English name into a phonetic representation using the Festival speech synthesis system. Second, translate the phoneme sequence into a sequence of generalized initials and finals (GIFs) basically based on sub-syllables. Third, the GIF sequence is transformed into a sequence of Pin-Yin symbols and finally, translation of the Pin-Yin sequence to a Chinese character sequence [12].

Ravi and Knight [74] proposed a method for automatic transliteration without any parallel resources. The transliteration task was addressed as a deciphering problem in four-stage cascade of weighted finite-state transducers [67]. They showed that it is possible to learn cross-language phoneme mapping tables using only monolingual resources text.

Dhore *et al* [75] proposed a direct (forward) machine transliteration model for Hindi and Marathi (Devanagari script languages) NEs to English. The proposed statistical phonetic approach used phonemes and the length of NEs as features for supervised learning. They segmented a given word into a number of transliteration units (TUs) based on linguistic knowledge and then transliterated each unit using a phonetic map table for Devanagari to English. They considered 15224 NEs having length varying from 2 to 8 TUs (*akshara*) for evaluation and found the transliteration accuracy average to be more than 90%. Also, it was observed that the accuracy is inversely proportional the number of TUs [75]. Using the phonetic-based model leads to significant improvement for the NEs consisting of 4–8 *akshara*, since Marathi and Hindi are phonetically rich languages. The issue with this model is that when the NE consists of multiple smaller length sub-NEs, the transliteration accuracy decreases.

Grapheme based

Li *et al* [10] proposed a direct orthographic mapping (DOM) model to generate transliterations from English to Chinese. The model allows DOM between two languages through a joint source-channel model (n-gram transliteration model). With the n-gram model, the orthographic alignment process is automated to derive the aligned TUs from a bilingual dictionary. The model under the DOM framework gained large improvement in transliteration accuracy over state-of-the-art machine learning algorithms [10]. The model can be applied to other language pairs such as English–Korean and English–Japanese. This is possible since bilingual alignment is incorporated into the decoding process in the n-gram transliteration model, which allows achieving a joint optimization of alignment and transliteration automatically.

Malik *et al* [39] proposed a Hindi–Urdu transliteration model using Finite-State Transducers (FST). The features of the FST were exploited in the model to obtain Hindi–Urdu transliteration using generic and flexible Universal Intermediate Transcription (UIT) scheme. The UIT encoding scheme was used to encode natural language to ASCII uniquely. The model was tested on Hindi–Urdu transliteration; it was observed that it is very efficient for inter-dialectal languages as well. The authors have also introduced UIT for the same pair on the basis of their common phonetic repository in such a way that it can be extended to other languages like Arabic, Chinese, English, French, etc. [39]. Three issues were found while transliterating Urdu terms to their equivalent Hindi terms. First, **multi-equivalences**: an Urdu character *Sheen* has two equivalents 'श' and 'ष', which are not distinguished and hence produced either of them. Second, **no equivalence**: ('ङ', 'ञ' and 'ण' do not have any equivalent character in Urdu. And third, **missing diacritical marks**. Similar issues were observed in Hindi–Urdu script conversion. The characters

'त', 'ह', 'स' and 'ज' have multiple corresponding Urdu characters.

Chinnakotla *et al* [29] proposed a rule-based system for the transliteration of Hindi to English, English to Hindi and Persian to English languages pairs. The model accomplished the transliteration task in three steps. First, word origin was identified using Character Sequence Modelling (CSM) on the source side. Second, transliteration candidates were generated using rule-based manual character mapping and finally, transliteration candidates were ranked using CSM on target side. They conclude that the rule-based system is the only viable option for the resource-scare language pairs [29]. The proposed system can be useful for majority of the languages because rule-based system is suitable for phonological writing system and most of the languages are phonological. The system may not transliterate terms properly to an over-determined writing system (multiple letters used for one sound), and from an undermined writing system (one letter used for multiple sounds).

Zhang *et al* [76] developed a model that introduced two pivot strategies for statistical machine transliteration, namely system-based pivot strategy and model-based pivot strategy. Given two independent source–pivot and pivot–target name pair corpora, both the models learn for generating transliteration. While the model-based strategy aims to learn direct source, target model by combining the two individually learned models on source–pivot and pivot–target corpora, the system-based strategy learns two joint source-channel models, i.e., a source–pivot model and a pivot–target model. Experimental results on benchmark data showed that the system-based pivot strategy is effective in reducing high resource requirement of training corpus for low-density language pairs [76]. A system-based model is very effective for capturing the phonetic information of source language, reducing the error propagation and enhancing transliteration performance by generating more pivot elements. The model-based strategy performs poor when English is not involved as one of the languages compared with the language paired with English. This happens mainly due to lesser availability of training data.

Kumaran *et al* [77] proposed a combination of two compositional machine transliteration systems, namely serial and parallel. The serial compositional system chains individual transliteration components, say, $X \rightarrow Y$ and $Y \rightarrow Z$ systems, to provide transliteration functionality, $X \rightarrow Z$. The parallel composition evidence from multiple transliteration paths $X \rightarrow Z$ is aggregated for improving the quality of a direct system. They showed the functionality and performance benefits of the compositional methodology using a state-of-the-art machine transliteration framework in English and a set of Indian languages, namely, Hindi, Marathi and Kannada. They also demonstrated the utility and practical aspect of compositional approach by integrating with compositional transliteration systems with a CLIR system. In the error analysis it was found that approximately 60% of the errors were due to incorrectly transliterated vowels – a phenomenon obvious in phonological languages. The transliteration performance can be significantly improved by proper handling of the vowels.

Wang *et al* [78] proposed a grapheme segmentation approach for English–Korean and English–Chinese pairs. The approach completes the works in four steps: preprocessing, alignment, DirecTL+ training and results re-ranking. The grapheme-level alignment was performed using the M2M aligner with some handwritten rules. The DirecTL+ was used for the training of source–target grapheme pairs. As a number of models are trained, for the re-ranking, orthography similarity ranking and Web-based ranking have been used [78]. For the English–Chinese on NEWS-2011 and -2012, *F*-scores were 67.19% and 64.55%, respectively. The Web corpora here contains almost actual usages of the transliterations. Hence, Web-based ranking method has significantly improved the transliteration performance.

Hybrid approaches

Oh and Choi [79] proposed an ensembled grapheme and phoneme (correspondence-based model)-based transliteration model for English to Korean. The model first maps source graphemes to source phonemes using a standard pronunciation dictionary and then using both source graphemes and phonemes, target graphemes are generated. Three different machine learning approaches (MaxEnt, Decision Tree and Memory-Based Learning) were used for source–target transliteration.

Oh and Choi [80] extended the work using an ensemble of three different transliteration models with the perception that one transliteration model alone has limitation on reflecting all possible transliteration behaviours. Several transliteration models, complementary in nature, were used in order to achieve a high-performance machine transliteration system. The method also used transliteration ranking with the help of Web data and relevance scores from different transliteration models. They report evaluation results for ensemble transliteration model and experimental results for its impact on IR effectiveness. Machine transliteration was tested on English-to-Korean transliteration and English-to-Japanese transliteration and achieved 78–80% word accuracy [80]. The ensemble of three models makes a machine transliteration system possible to produce maximum number of correct transliterations by considering different ways of transliteration. Along with that, use of Web-based ranking and relevance score filter out the noisy transliterations.

The work by Oh and Choi actually widened a new direction of research on transliteration started by Al-Onaizan and Knight [24] and later Bilac and Tanaka [81], of combining several sources and/or including supplemental transliterations. This has led to several recent works such as Kumaran *et al* [77], Bhargava and Kondrak (2011, 2012) and Yao and Kondrak [82].

Wang *et al* [83] proposed a hybrid model for NE transliteration for the English–Chinese and Chinese–English. They attempted to solve the transliteration as translation problem and used a log-linear model to obtain an optimal one from the possible result [83]. Expansion of the training set with the help of Wikipedia in the NEs increased the evaluation score.

Combined approaches

Huang [84] proposed a cluster-specific transliteration model for NEs. He grouped name origins into a smaller number of clusters based on language of origin (i.e., the same set of letter(s) may have different pronunciations in different languages). The authors then trained transliteration and language models for each cluster under a statistical MT framework. Given source words were classified into the most likely cluster; later, using corresponding models, transliteration would be performed. The process of transliteration is completed in four steps. First, segmentation of a Chinese character sequence into a source phrase sequence; second, convert these Chinese characters into their romanization form (Pin-Yin), then align the Pin-Yin with English letters using phonetic string matching; third, identify the initial phrase alignment path based on the character alignment path; and finally apply a beam search around the initial phrase alignment path, searching for the optimal alignment that minimizes the overall phrase alignment cost [84]. Transliteration of NEs under Chinese cluster achieves 90% accuracy, which is the highest compared with other language clusters used in the experiment. Incorrect classification of language origin leads to variation in evaluation scores.

Huang's work introduced a new dimension that transliteration can have many linguistic origins, which spawned a body of work in the following years. Various name classification methods were proposed using supervised and unsupervised techniques li2004joint [13, 60, 84–86].

Finch *et al* [13] applied Recurrent Neural Network (RNN) language model to the joint source-channel model for transliteration generation. The transliteration was performed in a three-stage process: source to target graphemes alignment training using Bilingual Bayesian grapheme model; transliteration generation using phase-based SMT combining a joint source-channel model, a target language model, a grapheme insertion model and a grapheme sequence insertion model; and re-scoring using MaxEnt model and RNN language model [13]. Incorporation of RNN in the re-scoring process could not improve the scores significantly.

Josan and Lehal [87] used an approach for transliteration from Punjabi to Hindi. The approach combined a basic character to character mapping technique with rule-based and Soundex-based enhancements [87]. Subsequently Josan and Kaur [88] enhanced their system based on the noisy channel model:

$$t_{best} = \underset{t}{\operatorname{argmax}}\{Pr(t|s)\}.$$

This formula was reformulated using the Bayes rule, where formulation allowed for a target language letters' *n*-gram model $p(t)$ and a transcription model $p(s|t)$. Given a sequence of letters s, the *argmax* function is a search function to output the best target letter sequence. Experimental results show that their approach effectively improved the word accuracy rate and reduce the average Levenshtein distance between generated words and target words by a large margin from the baseline [87]. The baseline approach has low accuracy because of three issues: multiple equivalent letters for a single letter (sound [ʃ] in Punjabi has two equivalent letters in Hindi 'श' and 'ष'), single letter equivalent of multiple letters and no equivalent letter in the other language (Hindi letter ऋ has no corresponding letter in target language). However these issues were partially resolved using phonetic rules on output of baseline approach.

Khapra *et al* [89] have proposed a low-cost Quality Control (QC) mechanism that may be applied on transliterations collected by crowd-sourcing. Huge transliteration pairs are needed for training state-of-the-art transliteration engines that is infeasible for collection using experts. Hence crowd-sourcing could be a cheaper alternative, provided a good QC mechanism is in place. The approach lies on a rule-based 'Transliteration Equivalence' approach that takes as input a list of vowels in the two languages and a mapping of the consonants in the two languages. Their experiments suggest that validation of transliteration pairs through crowd sourcing is a viable alternative [89]. However, a related issue was that the crowd had incorrectly transliterated some non-Indian origin words. They were orthographically incorrect but phonologically correct. This happened because vowels had an ambiguous phoneme–grapheme mapping in English. Hence, non-native speakers could not distinguish vowel variations.

4.2 *Transliteration mining*

Transliteration pairs are often mined from parallel or non-parallel, comparable corpora or the Web. The *parallel corpora* are collection of texts in two or more languages. Aligned texts are exact translations of one language into one or more languages. The *comparable corpora* are also a collection of texts in two or more languages. Here the texts are similar, meaning that they contain similar information, but are not exact translations of each other. A number of techniques have been proposed for transliteration mining. These techniques are based on the ways mining is being done. They may be categorized into three types. In the article, extraction is also used at mining place, first, based on *phonetic similarity*; second, using *machine learning* techniques and the third, based on *word co-occurrences*.

4.2a *Phonetic similarity:* This is the way of finding target word w'_1 written in native script for a given source word w_1 written in foreign script. For example, assume 'Dhanbad' is a source word w_1 and we have the parallel or comparable corpora containing words aligned like 'Dhanabad 'धनबाद'', where the first word 'Dhanabad' is w_2 and aligned word 'धनबाद' is transliteration of w_2, represented as w'_2. Approaches based on phonetic similarity measure the similarity between w_1 and w_2 in the parallel or comparable corpora and extract the transliteration w'_2 as the closest candidate for w'_1. A number of schemes have been used for similarity measure; most of them are derived from different edit distance techniques such as Levenshtein's Distance (LD) algorithm, Longest Common Subsequence (LCS) algorithm and Jaro–Winklor distance algorithm. Edit operations (insertion, deletion and substitution) in these techniques require some weight (*fixed* or *variable*) assignment. Assigning variable weights to edit operations depending on the characters, one can design similarity schemes that are more sensitive to a given task. These variable assignments can be divided into two main groups. One approach is to manually design edit operation weights on the basis of linguistic intuition and/or physical measurements. Another approach is to use machine learning techniques to derive the weights automatically from training data composed of a set of word pairs that are considered similar [90]. Some of the salient papers are discussed here.

Kuo and Yang [91] proposed a model to build transliterated term lexicons from the Web. While constructing the confusion table for syllable–phoneme cross-lingual conversion they applied a simple syllable alignment algorithm. The simple syllable algorithm accepts equal number of syllables to align syllable–phoneme cross-linguality and generates a mapping table for source and target languages. Two conversions, phoneme–phoneme and text–phoneme, were applied for syllabification to calculate the degree of similarity between phonemes for transliterated term extraction [91]. The cross-linguistic relation constructed using Text-Syllable Algorithm (TSA) and Text-Phoneme (TP) is called CTP, and the cross-linguistic relation using phoneme–syllable algorithm (PSA) and phoneme–phoneme (PP) is called CPP. Extraction based on syllable conversion in TSA outperformed phoneme conversion in CPP mapping. It was achieved at the final stage when the quality of generated syllable combinations improved. A similar case was observed in the CTP mapping. In general, CPP performed better than CTP in terms of the extracted term pairs number, because the combinations found using TSA are larger than those found using PSA.

Subsequently, Kuo and Yang [92] proposed another approach to take care of pronunciation variation issues. The confusion matrices generated by automated speech recognition (ASR) have been a basis for both improving pronunciation variation and constructing cross-linguistic syllable and phoneme conversions. The use of ASR improved the mining performance gradually by using cross-linguistic syllable-phoneme confusion matrices trained and refined progressively from mined term pairs. Many terms extracted in the experiment are new to the existing lexicons. Experiments on mining information from the extracted pairs were also conducted. The experimental results showed that taking pronunciation variation into account did make extraction of paired cognates more effective [92].

Kuo *et al* [93] proposed a model for extracting transliteration pairs from Web corpora. The model completed the entire process in four steps. Step 1 identifies English words from a given Chinese sentence. Step 2 decides the candidacy of transliteration in close context of English words. Step 3 converts English-Chinese candidates to syllables, English syllables and Chinese syllables, and identifies the most probable (using k-neighbourhood) Chinese syllables that match using the phonetic similarity model. They formulated a machine transliteration process using a syllable-based phonetic similarity model that consists of orthographical confusion matrices that established mappings between Romanized syllable codes and their Chinese character. Both n-gram supervised and unsupervised machine learning approaches were used to measure phonetic similarity. Step 4 validates/qualifies the Chinese syllable through hypothesis test and accepts corresponding Chinese word as transliteration of identified English word. They validated the system and achieved an F-measure of 0.739 with supervised learning [93]. Phonetic similarity measure (PSM) offers multiple transliteration variants for each English word since prior knowledge of romanization for a Chinese name helps in direct orthographical mapping. Overall, the recall rate is low for the foreign words of Korean and Japanese origins. This is probably due to the fact that romanization rules of Korean and Japanese names were not adequately captured in a PSM model that was mainly trained on English–Chinese pairs.

Oh and Isahara [94] proposed a transliteration lexicon acquisition model that mines the Web for transliteration lexicons. They applied forward and backward (Joint) candidate validation techniques for finding English–Japanese and English–Korean transliteration candidate pairs from the Japanese, and the Korean Web, respectively. Phonetic-similarity was used for comparison to recognize transliteration pair candidates on the Web and finding the correct transliteration pairs. They experimentally proved that their techniques effectively found transliteration lexicons on the Web [94]. The experimental results showed that joint-validation model validates transliteration pairs more effectively than the forward-validation model alone. Bilingual Phrasal Search (BPS) produces more reliable results than Bilingual Keyword Search (BKS) and Monolingual Keyword Search (MKS), because joint-validation model based on BPS is very effective in validating transliteration pairs.

Oh *et al* [95] focussed on the validation of transliteration pairs using updated corpora, such as the Web. They proposed a hybrid model (combining several models) for transliteration pair acquisition. First, transliteration pair (TP) candidates are extracted based on transliteration boundaries and then they are validated using three validation models (Corpus-based Similarity Model (CSM), Phonetic Similarity Model (PSM) and Phonetic Conversion Model (PCM)). It was experimentally shown that a hybrid model was more effective than an individual transliteration pair acquisition model alone [95]. The model fails to distinguish the relevant and irrelevant pairs from candidate TPs when they differ by one or two syllables. Documents were rarely searched using the phrasal search when the sizes of candidate TPs were long; thus, the CSM cannot distinguish relevant TPs from irrelevant TPs.

Lee *et al* [96] proposed a model for English–Chinese transliteration pair extraction from parallel corpora. The model can extract bilingual name and transliteration pairs. For extraction, parallel texts were first aligned at the sentence level. Then, the proper nouns were identified in the source text. After this, the transliteration model is applied to the processed text. In the transliteration phase, identified word was decomposed into TUs, i.e., source TUs. Then the source TUs were mapped with target TUs from all possible aligned candidates. Alignment path having the maximum accumulated log score among all possible alignment was selected. The parameters of the proposed model were automatically acquired through statistical learning from a bilingual proper name list. They reported how the model was applied to extract proper names and corresponding transliterations from parallel corpora. Experimentally achieved word and character precisions were 93.8% and 97.8%, respectively [96]. The model did not require any additional data like pronunciation dictionary or source words grapheme–phoneme rules to learn the parameters; rather, it learned automatically from bilingual name pairs. The framework can be extended to other language pairs if there exist transliteration training data. The proposed method does not require any intermediate process other than matching TUs of two languages.

Sproat *et al* [97] proposed two distinct models for Chinese–English name entities transliteration using comparable corpora. For a given English name from the comparable corpora, first all candidate Chinese character *n*-grams are identified belonging to the same time window; later each such candidate is assigned a score based on how likely the candidate is to be a transliteration of the English name; finally, the scores of all the candidate transliteration pairs are propagated globally based on their co-occurrences in document pairs in the comparable corpora. Scores were calculated based on pronunciation and frequency correlation. They stated that both the approaches worked quite well individually; however, combining them can improve the result even further [97]. Mean Reciprocal Rank (MRR) scores for the core English names took a quantum leap compared with those for all the English names. However, it missed many names that do not necessarily occur in the comparable corpora. The authors state insufficient data in the training set as one of its possible reasons.

Udupa *et al* [98] proposed the MINT (MIning Named-entity Transliteration equivalents) method for effective and scalable mining of NE transliterations from large comparable corpora and extended it in 2009. The algorithm is completed in two stages. First, alignment of transliteration pairs is done using a cross-language document similarity model to align multilingual news articles. Later, Named Entity Transliteration Equivalents (NETEs) are mined from the aligned articles using a transliteration similarity model. They have shown that the approach is effective on six different comparable corpora between English and four languages from three different language families [98, 99]. MINT performs close to optimal on pairs of similar articles when the pairing of news articles is known in advance. This condition puts a limitation that scores will be affected when paired articles in two languages are not available.

Sato [100] proposed a non-productive machine transliteration for English–Japanese NE transliteration. The framework assumes that the candidate transliteration list includes the correct transliteration. Therefore, transliteration problem is attempted as a search/selection problem. Based on the rules, a set of source language letters is mapped to a set of target language letters (string). The algorithm is practicable, even with candidate lists consisting of over a million transliteration entries [100]. For Japanese–English back-transliteration, performance decreases when the larger candidate list is used. In this case more ambiguous solutions are obtained because the list contains more terms with similar spellings and sounds.

Transliteration variants may cause incomplete search results because using one transliteration as a query keyword may fail to retrieve the Web pages that use a different word as the transliteration. Hsu and Chen [101] proposed a framework for mining synonymous transliterations from the Web snippets. The results can be used to construct a database of synonymous transliterations, which can be utilized for query expansion, especially for incomplete search cases. The authors also showed that the proposed framework can effectively retrieve the set of snippets during IR by considering synonymous transliterations. It was found that inclusion of context information helped improve the ranking of synonymous transliterations extracted. While removing the standard words from the source that do not need to be transliterated, there is a chance of missing some synonymous transliterations if they are same as the standard words.

4.2b *Machine learning:* In this type, variable weight of operations (*insertion, deletion and substitution*) is assigned using machine learning (supervised and unsupervised) techniques to obtain the target word from a given source word. Some of the techniques include Lee and Choi [102], Klementiev and Roth [16], Sherif and Kondrak [103], Chen and Hsu [50], Goldwasser and Roth [104], Kuo *et al* [105] and Li *et al* [59]. Interested readers can find their discussion in the pre-cursor survey by Karimi *et al* [48]. In the following, we discuss some more.

Chang *et al* [7] proposed an unsupervised constraint-driven learning algorithm for identifying NE transliterations in three different bilingual corpora: Chinese-English, Hebrew-English and Russian-English. Their method did not require any annotated data or aligned corpora. Instead, it was bootstrapped using simple resources like a romanization table and a set of language-specific constraints. Using the table and/or constraints, transliteration pairs were identified. In the evaluation, they found that constraint-driven learning can significantly outperform existing unsupervised models and achieve competitive results compared with existing supervised models [7]. Using romanization in conjunction with constraints improved results dramatically, which shows that romanization table contains enough information to bootstrap the model when used with constraints.

El-Kahky *et al* [6] developed a model for transliteration mining using graph reinforcement. The generative model was used to infer source and target character sequence mapping. An initial set of mappings are learnt through automatic alignment of transliteration pairs at character sequence level. Later, these mappings are modelled using a bipartite graph. A graph reinforcement algorithm is then used to enrich the graph by inferring additional mappings. During graph reinforcement, appropriate link re-weighting is used to promote good mappings and to demote bad ones. The enhanced transliteration mining technique is tested in the context of mining transliterations from parallel Wikipedia titles in four alphabet-based language pairs, namely English–Arabic, English–Russian, English–Hindi and English–Tamil. Their results show that the approach was able to outperform the state of the art. They said that mining of transliterations from comparable or parallel text can enhance NLP applications such as MT and CLIR [6]. Graph reinforcement can help match phonetic variations within the same language, which can be applied in spelling correction and in transliterated IR.

Kaur and Josan [106] addressed the transliteration issues from English to Punjabi using a probabilistic approach. The approach includes supervised learning of aligned NEs in English and Punjabi using some statistical MT tools (MOSES, GIZA++, SRILM) followed by handcrafted transliteration rules to get the transliterations. The best word accuracy for the system reported was 63.31% [106]. Applying some transliteration during post-processing improved the system result. However, the schwa deletion algorithm did not help much in improving the system. This algorithm was designed for Hindi and it is not necessary that rules that work for Hindi will also work for Punjabi.

Dasgupta *et al* [40] proposed a back- transliteration system for English–Bengali. They used a parallel corpus of 83000 English–Bengali bilingual words. The English words were phonetically analysed using grapheme–phoneme (G2P) converter to generate TUs according to the technique of Ekbal *et al* [41]. Bengali words are segmented using linguistic rules. The alignments were done using two different computational models, namely, the joint source-channel model and the tri-gram model, to automatically identify and extract TU pairs from both the source and target language words [40].

Fukunishi *et al* [107] proposed a model to extract the transliteration words pairs. Wikipedia (English and Japanese) is used as a resource. They performed Bilingual Forced Alignment on Web Data using a Dirichlet process model (Bayesian nonparametric method) as it allows reuse of parameters. Features were then extracted from co-segmented text obtained during the alignment. Using a threshold value of the features, texts were classified into negative examples and positive examples (seed sentences). Using these extracted features (negative examples and positive example), a support vector machine (SVM) was trained to classify the correct and incorrect transliterations in candidate transliteration. After this, the trained SVM was used to classify the good and bad transliteration pairs on test data with the assumption that the correct transliteration pairs would be well arranged and generated easily, whereas incorrect pairs would be costly and generated in a more random manner in character. The approach works on the principle that generation using only grapheme sequence pairs that are in the model results in a high probability derivation. The features extracted from the alignment of the test data were not only based on the scores from the generative model but also on the relative proportions of each sequence that are hard to generate. The features are used in conjunction with an SVM classifier trained on known positive examples together with synthetic negative examples to determine whether a candidate word pair is a correct transliteration pair. They used training and test data from 2010 Named-Entity Workshop (NEWS10) tracks and use the performance of the best system for each language pair as a reference point. Their results show that the proposed features are powerfully predictive [107]. The approach is simple and does not depend on language-specific information about language pair involved and it will operate on the native script of languages grapheme sequences directly.

Sajjad *et al* [108] proposed a model to automatically extract transliteration pairs from parallel corpora in a language-pair-independent manner. Transliteration pairs are mined in both unsupervised and semi-supervised settings consistently. The authors modelled transliteration mining as an interpolation of transliteration and non-transliteration sub-models. Scores showed competitive performance on NEWS 2010 data. The approach is efficient and independent of language pairs. A number of false positives were decreased using seed data by considering close transliterations as incorrect ones. However, this approach excludes the possibility of close transliterations getting qualified even with minor spelling variations – a phenomenon that generally occurs during transliteration, and therefore, acceptance.

Another paper by Htun *et al* [109] explored integrating human expert knowledge with machine learning approaches for determining phonetic similarity of word pairs. The system consisted of a human to provide a structure for the edit costs that are based around a phonetically motivated model of phoneme sound groups, and a machine to determine precise values for these costs within two different frameworks based on stochastic edit distance: a method based on one-to-one expectation maximization (EM) alignment and a Bayesian many-to-many alignment approach. Experiments on Myanmar-English data show that human expert knowledge improves transliteration mining performance.

Durrani *et al* [34] proposed an unsupervised model for transliteration mining as a combination of two sub-models: a non-transliteration sub-model and a transliteration model. The transliteration sub-models learn character alignment based on the EM. In the fully unsupervised and language-independent method they tested on 7 different language pairs, namely Arabic, Farsi, Russian, Hindi, Bengali, Telugu and Urdu with English. They observed improvements from 0.23 to 0.75 (Δ 0.41) BLEU points across language pairs used in experiments. They also showed that their mined transliteration corpora provide better rule coverage and translation quality compared with the gold standard transliteration corpora [34]. The Gold Standard Transliteration (GST) system suffered from sparsity and did not provide enough coverage of rules to produce right transliterations. The Arabic words drop the determiner (al), but such additions were not present in gold transliteration pairs. For example, an Arabic word (Gigapixel) leads to wrong transliteration 'algegabksl'. Similar errors are observed in other language pairs as well.

Durrani and Koehn [35] proposed a model that improved *Urdu* $\rightarrow$ *Hindi* $\leftrightarrow$ *English* MT through transliteration and triangulation. First they construct an *Urdu* $\rightarrow$ *Hindi* SMT system by inducing triangulated and transliterated phrase-tables from Urdu-English and Hindi-English phrase translation models. Then they use it to translate the Urdu part of the Urdu-English parallel data into Hindi, thus creating an artificial Hindi–English parallel data. The phrase-translation strategies give an improvement of up to +3.35 BLEU (BiLingual Evaluation Understudy) points over a baseline *Urdu – Hindi* system. The synthesized data improve *Hindi – English* system by +0.35 and *English – Hindi* system by +1.0 BLEU points [35]. The approach suffers from data sparsity problem – the most common issue for the resource-scare languages.

Kunchukuttan and Bhattacharyya [110] proposed a phrase-based model to transliterate the Indian languages (namely Hindi, Bengali and Tamil) term to its native script. Character one-gram and bi-grams were used to train transliteration, and transliteration was mined using a module in Moses. Adding boundary markers helps in significantly improving the evaluation scores (MRR and transliteration accuracy). This improvement has been achieved probably due to reduction of errors like (i) missing initial vowels, (ii) wrong consonant in the first and last syllables and (iii) incorrect generation of *halanta* (an inherent vowel suppressor) character.

Shao *et al* [111] built a phrase-based transliteration model for English–Chinese and Chinese–English with the Moses and used M2M-aligner for TU alignment. Further, they used a multilingual re-ranking model to rank the transliteration extracted where the scores of translation models were used as features. The multilingual re-ranking model used here outperformed the baseline approach. Transliteration without language source identification is difficult because phonetic systems of languages are different. Hence, sometimes, it is impossible to get correct transliterations due to pronunciation differences.

Finch *et al* [112] proposed two models to obtain transliteration: first, a neural network model and second, a phrase-based SMT. For both the models, sequence alignment is done using a non-parametric Bayesian model. Scores of the neural network model are used in the phrase-SMT model, which improves the results [112].

4.2c *Word co-occurrence:* Assume that two content words *w* and *w'* are in the same document; generally, they are topically related. With this notion of co-occurrence, how near or far away from each other they are in the document is irrelevant, as is their order of appearance in the document. Some extraction models based on word co-occurrences include [9, 80, 113–115], which are discussed in [48].

Wu *et al* [116] proposed an English–Korean transliteration system for the Named Entity Workshop (NEWS). The model has two components: first, letter–phoneme alignment using an m2m-aligner[2] and second, a DirecTL-p[3] transliteration training model. Two re-ranking methods were used to select the best transliteration among the prediction results from the different models. One re-ranking method is based on the co-occurrence of the transliteration pairs in the Web corpora and the other one is the JLIS-re-ranking method, which is based on the features from the alignment results. Their runs achieve 0.398(standard) and 0.458(non-standard) in top-1 accuracy in the generation task [116]. Among the re-ranking methods used, Web-based ranking performed better than JLIS re-ranking. The reason was that

[2]The m2m-aligner is an algorithm that can be applied in letter–phoneme conversion, https://code.google.com/p/m2m-aligner/.

[3]The DirecTL-p is an online discriminative training model for string transduction problems and can be used for name transliteration and grapheme–phoneme conversion tasks, https://code.google.com/p/directl-p/.

Web-based ranking was based on the huge number of co-occurrences of transliterations in the Web corpora.

Qu [117] described a novel bottom-up approach for English–Chinese name transliteration that allows DOM between the two languages. The approach accomplished the task in three steps. First, a neighbourhood of locally relevant transliteration names is made using a latent semantic analysis of the appropriate grapheme form; second, those names in neighbourhoods are aligned via locally optimal sequence alignment. Finally, the maximum likelihood estimate is computed for every position to obtain probable Chinese transliterations of the given English name. The experimental results confirm its effectiveness in English–Chinese name transliteration [117]. It was observed that some English names were incorrectly transliterated as Chinese names. This is due to sequence alignments in transliteration that considered only grapheme information instead of using both grapheme and syllabic information. This kind of errors will have a great impact on overall system performance.

4.3 *Fusion approaches*

There are a few approaches that use both the techniques of transliteration generation and transliteration mining.

Zhao *et al* [5] proposed a HMM-based framework to transliterate NEs. The framework considered features from letter alignments and letter *n*-gram pairs learned from available bilingual dictionaries. Letter-classes, such as vowels/non-vowels, were integrated for further improvement in transliteration accuracy. The proposed transliteration system was tested on OOV NEs in statistical machine translation (SMT), and a significant improvement over the traditional transliteration approach was obtained. Furthermore, by incorporating an automatic spell-checker based on statistics collected from the Web search engines, transliteration accuracy was further improved. The proposed system was tested on their SMT system and applied to a real translation scenario from Arabic to English [5]. Multiple alignment features in block-extraction approach helped achieve significant improvement in accuracy. Adding spell-checking based on occurrence statistics obtained from the Web gave additional improvement in transliteration accuracy.

Oh and Isahara [68] proposed a method to improve the machine transliteration using multiple transliteration hypotheses and re-ranking them. Seven machine-transliteration engines were constructed to produce a set of transliteration hypotheses. Later, the hypotheses were re-ranked to select the best hypothesis. The proposed re-ranking method utilizes confidence-score, language model and Web-frequency features and combines them with machine-learning algorithms including SVMs and the maximum entropy model. During the evaluation they found that combining multiple schemes gives far better results than any individual ones [68]. The improvement in scores was achieved due to the re-ranking of transliterations produced by individual models. Re-ranking models used here extend the scope of a large number of candidate transliterations for ranking.

Chinnakotla *et al* [30] proposed a CLIR system for Hindi–English and Marathi–English as a part of CLEF Ad-Hoc Bilingual task, where several techniques, including transliteration generation and and, to some extent, transliteration extraction, are fused together. Out-of-dictionary words were transliterated using a rule-based transliteration approach. The resultant transliteration was then compared to the unique words of the corpus to return the '*k*' most similar words to the transliterated word. The resulting multiple translation/transliteration choices for each query word are disambiguated using an iterative page-rank style algorithm based on term–term co-occurrence statistics [30]. However, a limitation with the system is that rule-based method may generate incorrect transliterations when there is spelling variation in source word (OOV). The incorrect transliteration will degrade the performance of the retrieval system.

Compression Word Format (CWF) is a method for comparing minimal consonant skeletal forms of source and target words. This translation model does not generate target NEs; hence, it fails to map OOV words. To overcome this problem, Narasimhulu *et al* [118] proposed a model for mapping and generation with dynamic learning. The process in the proposed model was to compress the given source name and compare to the target database names. The model accurately maps the source name with the target database names; if the source name has a right equivalent target name then it retrieves the equivalent and relevant target names. Otherwise, it transliterates the source name and retrieves the relevant target names from the Web. Later, new words need to be updated in the database automatically. To automate the updation, a dynamic learning algorithm was designed and deployed. The accuracy of the proposed system was improved using a learning algorithm when compared with the manual updating process.

5. Transliterated search

Machine transliteration developed as an area of research with the advent of automatic MT. Primarily, NEs and other OOV words are transliterated from one script to another. However, with the rapid increase in the proportion of non-English speakers among the online users, transliterated search is on steady rise. Users search for the information related to their locality or the information written in their local language, but they query in the transliterated domain. Since most of these users are well conversant in two or more languages, often the language of documents or the script used in the documents does not matter. For example, in a multi-lingual country like

India, people sometimes write queries using Roman script or in two or more scripts within a single query, even though they look for documents in Hindi or English or in some local language (Assamese, Bengali, Kannada, Marathi, Odiya, Tamil, Telugu, etc.) either in native script or in Roman script depending on the knowledge of the user. This reality has opened up a new search domain of multi-script multi-lingual IR. Keeping this need in mind, Microsoft Research India (MSRI) introduced a track '*Shared task on Transliterated Search*' at the Forum for Information Retrieval Evaluation (FIRE)[4] in 2013 [119, 120].

Initially there were two subtasks: (1) query word labelling and (2) multi-script ad hoc retrieval for Hindi song search. Subtask 1 comprises classification and labelling of terms and NEs based on their languages (Hindi and English) followed by transliteration of Hindi terms. In FIRE-2014, NE recognition was added to this sub-task. Subtask 2 is retrieval of Hindi song lyrics, i.e., retrieval of mixed script documents where documents are in Roman, Devanagari and mixed script (both Roman and Devanagari). Queries were also given in the same format.

The development corpus consisted of 62,888 documents of Hindi lyrics written in Roman, Devanagari or in mixed script. The development set contained 25 queries and their relevance judgments. Queries are written in transliterated (Roman) form or in Devanagari script of Hindi song titles or some part of the lyrics.

In 2013, totally three teams (NTUN, GU and TUVal) participated in Subtask 2. Team NTUN (NTUN-3) considered three different types of queries: one, all query terms written in Roman; two, all queries completely transliterated into Devanagari and three, mixed script with all Hindi terms transliterated into Devanagari script. All the given queries were converted into these categories and then the Lucene IR system was used with term frequency-inverse document frequency (TF-IDF) term-weighting for retrieval [120, 121].

Team GU (GU-1) took a syllable-based approach for transliteration during query processing. They submitted two runs with two different query sets. The first one contained queries in both Roman and Devanagari while the second contained queries in Roman only. For retrieval, they used Language modelling with TF-IDF-based term-weighting. The former performed better than the latter [120, 122].

Team TUVal (TUVal-2) used word 2-grams for the indexing of song collections and applied a word bi-gram variant of TF-IDF-like models for retrieval (i.e., DFR). In query formulation, first they found inter/intra-script variants and then formulated the query as word-2 grams in the corresponding scripts taking into account the variants. In total they submitted three runs by varying parameters (min_len, Θ and *algo*) values. The min_len is the minimum length of terms considered for variation look-up to expand the query, Θ is the threshold for similarity measure and algo is an algorithm for ranking relevant documents. Parameters were set for Run-1 (2, 0.95, *TF-IDF*), Run-2 (2, 0.95, XSrqA_M) and Run-3 (3, 0.95, XSrqA_M), where Run-2 performed the best in FIRE-2013 [38, 120].

In FIRE-2014, four teams participated: BIT (BIT-2), BITS-Lipyantran (BITS-Lipyantran-2), DCU (DCU-1) and IIITH. The team BIT completed the subtask in three modules: document indexing, initial query formation, query term extraction and expansion. For document indexing and retrieval, Lucene was used. In initial query formation, the script of query terms is identified. For Roman script, the terms are transliterated into Devanagari using the Google transliteration API. For Devanagari script queries they used a corpus of Roman–Devanagari transliteration pair aligned (from source data). Later, phrasal queries were formulated as word 2-grams from Devanagari and transliterated query terms [119, 123].

Team IIITH accomplished searching using a language model with query expansion. All the documents are transliterated uniformly in Roman script and are used in creation of a posting list with field-based term-weighting. For example, title of lyrics was given high weightage, then the first line of lyric, first line of each stanza line with specific singers name, etc. Finally, using TF-IDF metric term-document, the frequency count of posting list is normalized. Edit distance technique was used for query expansion. In retrieval, applying query expansion on test queries, top 15–20 variations were selected as seed values, which were further used to generate top-20 documents [119, 124].

The team BITS-Lipyantran deeply focussed on spelling variations and query expansion. First, they transliterated back the queries and documents into Devanagari script. Then they created sub-words by removing the vowels (matras) from words. These words are used for indexing (called sub-word indexing). Instead of multiscripts, they approached monoscript IR only. Their system performed the best in FIRE-2014 for transliterated search task [119, 125].

The team DCU completed retrieval in three phases: document indexing, automatic transliteration and retrieval. Before document indexing, queries were normalized using dictionary-based query expansion and rule-based character sequence normalization method. For the out-of-dictionary (English–Hindi transliteration paired dictionary) terms in queries, SMT was used for automatic transliteration. Expanded queries in both Roman and Devanagari scripts were used for retrieval [119, 126].

In the following section, we define different metrics relevant to transliteration and retrieval and then we compare performance of different techniques discussed so far.

6. Evaluation

Evaluation is an important issue in the domain of transliteration. First, we shall define some primary metrics that are borrowed from text search and retrieval [1] domain and then their customization in transliteration [127].

[4]http://fire.irsi.res.in/fire/.

Precision: Precision (P) is the proportion of relevant documents in the set of retrieved documents or items:

$$precision\ (P) = \frac{\#relevant\ items\ retrieved}{\#retrieved\ items}. \quad (1)$$

Recall: Recall (R) is the ratio of retrieved relevant documents from the set of relevant documents or items:

$$recall\ (R) = \frac{\#relevant\ items\ retrieved}{\#relevant\ items}. \quad (2)$$

Average precision: Average precision (AP) is the average of the precision values obtained for the set of top 'k' retrieved documents (R_k) calculated after each relevant document is retrieved. Here, the number of relevant documents for a given query is 'm':

$$average\ precision\ (AP) = \frac{1}{m}\sum_{k=1}^{m} precision(R_k). \quad (3)$$

Mean average precision (MAP): *MAP* is the average of *AP* for a set of information needs or queries (Q):

$$mean\ average\ precision\ (MAP) = \frac{1}{|Q|}\sum_{i=1}^{|Q|} AP(i). \quad (4)$$

Evaluation is an important issue in the domain of transliteration. Evaluation in the area mainly considers transliterated or target word's accuracy. Two different accuracy measures used are word accuracy and character accuracy. Karimi *et al* [48] have divided techniques of accuracy measures into two categories, namely *single-variant* and *multi-variant* metrics.

6.1 *Single-variant metrics*

The first metric *word accuracy WA*, also known as *transliteration accuracy or precision*, is defined as follows [120]:

$$transliteration\ precision\ (TP) = \frac{\#correct\ transliteration}{\#transliterated\ test\ words}. \quad (5)$$

More than one transliteration are possible. However, depending on requirement, cutoff can be set for top-*n* candidates or possible transliterations. If there is restriction for a single answer, the system can return the first candidate only as target and for top-5, first five candidate will be returned, whereas in this section, top-1 has been considered.

Transliteration recall (*TR*) is the ratio of correct transliteration to referenced transliteration defined later. The reference transliteration is the gold answer (number of relevant in terms of relevance) in reference file (or *qrels*).

$$Transliteration\ recall\ (TR) = \frac{\#correct\ transliteration}{\#reference\ words}. \quad (6)$$

Character accuracy (*CA*) measures the fraction of matched characters for word pairs:

$$character\ accuracy\ (CA) = \frac{length\,(T) - EditDist(T, L(T_i))}{length\,(T)} \quad (7)$$

where *length*(*T*) is the length of reference word *T*, $L(T_i)$ is ith candidate transliteration and *EditDist*() gives the edit distance between T and $L(T_i)$. The *CA* is dependent on *EditDist*() measure; increment in *EditDist*() decreases the *CA*, which shows increment in character mismatch.

Some extraction-based techniques use *F-score* measure as well:

$$transliterationF - score = \frac{2 \times TP \times TR}{TP + TR} \quad (8)$$

Here *TP* and *TR* are transliteration precision and transliteration recall, respectively.

MRR is also used for ranking a set of candidate transliterations, which is defined as follows:

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{R_i}. \quad (9)$$

Here, N is the total number of test words and R_i is the rank in a set of candidate transliterations L in which the ith test word has a correct transliteration.

6.2 *Multi-variant metrics*

Due to variation in corpus creation by different transliterators there is a possibility of multiple transliterations. These candidates should also be taken care of. Karimi *et al* [53] introduced three groups of word accuracies: *uniform, majority* and *weighted.*

In case of *Uniform Word Accuracy (UWA)*, equal weight is assigned to all the candidates and all transliterations are considered to be equally valid. In case of *Majority Word Accuracy (MWA)*, only one transliteration will be considered as correct based on majority voting. In case of *Weighted Word Accuracy (WWA)*, a weight is assigned to each of the transliterations based on the number of times they have been suggested by multiple transliterators. All transliteration variants are valid but with a given weight. BiLingual Evaluation Understudy (BLEU), a translation evaluation technique, is close to this category and used for transliteration evaluation.

The *BLEU* is based on modified precision and defined to measure the MT accuracy [128]. First, geometric mean of modified precision is computed, which is then multiplied by an exponential brevity penalty factor *BP*. For example, compute the geometric average of the modified *n*-gram precisions, P_n using *n*-grams up to length *N* and positive weights w_n summing to one. Next, let *c* be the length of the candidate translation and *r* be the effective reference corpus length. The *Brevity Penalty* (*BP*) and *BLEU* are defined in the following way:

$$Brevity\ Penalty\ (BP) = \begin{cases} 1, & c > r \\ e^{1-\frac{r}{c}}, & c <= r \end{cases} \tag{10}$$

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log P_n\right). \tag{11}$$

7. Comparison

In the earlier sections, we discussed different techniques of transliteration followed by a search in the transliterated domain. In the following subsections, we first compare the performances of different transliteration techniques and then of search techniques.

7.1 *Transliteration techniques*

As discussed in section 4, transliteration techniques can be of mainly two paradigms: generation and extraction, although there are some models that are combinations of both. In order to quantitatively compare performances of different transliteration methods/models, here we use standard performance metrics like precision, recall and *F*-score along with some other metrics discussed in section 6.

We have included the scores of models in tables 1, 2 and 3 for generative, extraction and fusion techniques, respectively. These models have used different training/development datasets for designing the transliteration system.

In table 1, we summarize different generative techniques with their respective performances. Three types of corpus, namely graphemes, phonemes and syllables, were used for transliteration mapping. Most of the techniques used probabilistic approach for source–target alignment. The joint source-channel model, extended Markov model, character sequence model, decision tree model, maximum Entropy model, memory-based learning model and MT model have been used so far for transliteration generation. Among the listed techniques in table 1, Chinnakotla *et al*, Zhang *et al* and Kumaran *et al* tested their system on the standard (Named Entity WorkShop (NEWS)-2009) dataset. There are variations in the scores of the models, but grapheme-based approaches show generally better performance, except for the character sequence model.

Table 2 shows the scores of various mining models. As discussed in section 3, the mining transliteration models are classified based on phonetic similarity, machine learning and word co-occurrence. Among them, approaches based on phonetic similarity performed well but approaches based on machine learning are also compared. The method based on word co-occurrence did not perform well compared with the other two approaches.

In table 3, we have listed some models that follow both the strategies discussed above to obtain transliterations. There were only a few works that we could find as far as fusion techniques are concerned. However, based on the scores reported, these techniques are not better than only generation and/or only extraction (mining) counterparts.

In all three tables, we have tried to summarize the recent works under different paradigms, the models and data used and the scores obtained. There were wide variations in the languages, data collection and the metrics. The comparison of two systems cannot be justified if the datasets and metrics are not the same. The scores might be biased with dataset, the source and destination languages, etc. It is, therefore, not possible to choose a single best technique. However, the techniques using the same dataset (e.g., NEWS 2010 or 2015) can be compared among themselves.

Overall scores suggest that mining-based approaches can be preferred over generative approaches. However, one needs to remember that these two are two different tasks altogether. Mining-based system can focus on precision over recall and therefore can be made very precise as per the needs of the application. This is not possible for generation-based techniques.

7.2 *Transliterated search techniques*

Table 4 shows scores (MAP and MRR) of retrieval approaches used by participants in FIRE-2013 and 2014. Most of them have applied *TF-IDF*-based retrieval techniques along with Language Modelling (LM). TUVal performed the best among the lot because of its rich query expansion based on deep learning strategy. Another competitive performance was exhibited by BITS-Lipyantran with its sub-word indexing approach.

8. Discussion

All the transliteration approaches have their own importance with intrinsic strengths and weaknesses. Also, their performance depends on the availability of the resources. The techniques discussed here were mostly published during the period 2009–2015. Although we tried to include papers from reputed journals and conferences globally, our focus was on the Indian languages. In this section, therefore, we would like to see them from a different perspective: belonging to non-Indian and Indian language categories.

Table 1. Performance of transliteration generation techniques.

Authors, year and language/ script	Corpus specification	Models	Metrics	Performance
Jung *et al* [20], English–Korean	Syllable	Extended Markov model	Recall	87.50%
Oh and Choi [21], English–Korean	Phonemes	Direct mapping model	Precision	71.71%
Virga and Khudanpur [12], English–Chinese	Phonemes	Statistical translation model	*MAP* (retrieval)	50.10%* 51.70%**
Ravi and Knight [74], English–Japanese	Phoneme	Cascaded finite state transducer model	Error rate	66.00%
Dhore *et al* [75], Hindi–English (H–E) and Marathi–English (M–E)	Phoneme	Statistical model (ML)	Recall@5 aksharas	90.60%
Li *et al* [10], English–Chinese (E–C), Chinese–English (C–E)	Grapheme	Joint source channel (*n*-gram TM) model	Error rate E–C C–E	29.90% 62.10%
Malik *et al* [39], Hindi–Urdu	Grapheme	Finite state transducer model	Precision	97.50%
Chinnakotla *et al* [29], Hindi–English (H–E), English–Hindi (E–H), Persian–English (P–E)	Grapheme [NEWS 2009 dataset]	Character sequence modelling (CSM)	*MRR* H–E E–H P–E	54.70% 45.90% 34.30%
Zhang *et al* [76], Chinese–Japanese (C–J), Chinese–Korean (C–K), Japanese–Korean (J–K)	Grapheme [NEWS 2009 dataset]	Joint source channel (*n*-gram) model	*F*-score direct (E–C) System based (E–C) Model based (J–E)	87.14% 87.14% 83.83%
Kumaran *et al* [77], English–Russian (E–R), English–Hindi (E–H), English–Turkish (E–T) and English–Kannada (E–K)	Grapheme [NEWS 2009 dataset]	Compositional model	*F*-score E–R E–H E–T E–K	92.70% 87.70% 89.80% 86.90%
Wang *et al* [78], English–Chinese (E–C) and English–Korean (E–K)	Grapheme [NEWS 2011 dataset] [NEWS 2012 dataset]	Compositional model	*F*-score E–C E–K	67.19% 64.55% 73.30% 76.14%
Oh and Choi [79], English–Korean (E–K), English–Japanese (E–J)	Hybrid	Decision tree (DT) Max. ent. model (MEM) MEM-based learning (MBL) DT MEM MBL	W.A. (E–K) W.A. (E–J)	62.00% 63.30% 66.90% 66.80% 67.00% 72.20%
Oh and Choi [80], English–Korean (E–K), English–Japanese (E–J)	Grapheme, phoneme and hybrid	DT MEM MBL DT MEM MBL	W.A. (E–K) W.A. (E–J)	73.20% 73.00% 78.90% 76.40% 77.20% 80.00%

**MAP* of retrieval systems without Named Entity transliteration.

***MAP* of retrieval systems with Named Entity transliteration.

8.1 *Transliteration for the non-Indian languages*

Enormous amount of work has been carried out for the foreign languages such as Chinese, Japanese, Korean and Russian. These works include transliteration as well as back-transliteration. Following observations can be drawn from the approaches in this context.

- It has been observed that by and large, works are for English to other languages (E–X).

Table 2. Performance of transliteration mining techniques.

Authors, year and language script	Corpus specification	Models	Metrics	Performance
Oh and Isahara, 2006 [94], English-to-Japanese (E-J), English–Korean (E-K), Japanese-to-English (J-E) and Korean–English (K-E)	Phonetic	Joint validation approach (PS)	Pair accuracy @TOP-5 E–J E–K J–E K–E	90.20% 90.10% 90.50% 88.70%
Oh *et al* [95], English-to-Korean	Phonetic	Hybrid model ((PS))	*F*-score	88.30%
Sproat *et al* [97], Chinese-to-English	Phonetic	Phonetic correspondence and word distribution models (PS)	All *MRR* Core *MRR*	96.50% 96.60%
Kuo *et al* [93], English-to-Chinese	Phonetic	Phonetic similarity (PS)	*F*-score	
Udupa *et al* [98], English-to-Kannada (E-K), English-to-Hindi (E-H)	NE equivalence	Mining equivalent Transliteration model (PS)	*MRR*@5 real Near ideal Ideal Real Near ideal	95.00% 94.00% 88.00% 95.00% 87.00%
Chang *et al* [7], English–Chinese (E–C), English–Russian (E–R) and English–Hebrew (E–H)	Orthographic	Unsupervised constrain-driven model (ML)	ACC *MRR*	73.00% 91.00%
El-Kahky *et al* [6], English–Arabic (E–A), English–Russian (E–R), English–Hindi (E–H) and English–Tamil (E–T)	Orthographic	Graph re-enforcement model (ML)	*F*-score E–A E–R E–H E–T	94.10% 92.30% 93.20% 95.50%
Dasgupta *et al* [40], English–Bengali	Phoneme and grapheme	Joint source-channel model (ML)	*MRR*	18.33%
Fukunishi *et al* [107], English–Arabic (E–A), English–Chinese (E–C), English–Hindi (E–H), English–Japanese (E–J), English–Russian (E–R) and English–Turkish (E–T)	Grapheme	Bayesian alignment approach (ML)	*F*-score E–A E–C E–H E–J E–R E–T	93.60% 56.30% 95.60% 92.50% 92.10% 93.40%
Durrani *et al* [34], English–Arabic (E–A), English–Russian (E–R), English–Hindi (E–H), English–Bengali (E–B), English–Telugu (E–T) and English–Urdu (E–U)	Orthographic	Statistical machine translation (ML)	*BLEU*	Δ 0.41
Wu *et al* [116], English–Korean	Orthographic and phoneme	Substring alignment and re-ranking model (WCO)	*MAP*	45.80%
Qu [117], English–Chinese	Orthographic	Latent analogy model (WCO)	W.A. C.A.	56.00% 88.00%

Table 3. Performance of transliteration fusion techniques.

Authors, Year and Language/ Script	Corpus Specification	Models	Metrics	Performance
Zhao *et al* [5], Arabic–English	Grapheme	Log-linear block transliteration model	Recall	72.16%
Oh and Isahara [68], English–Japanese (E–J) and English–Korean (E–K)	Phonemes	Used multiple transliteration engines and hypothesis re-ranking	Recall E–J Recall E–K	90.50% 89.70%
Chinnakotla *et al* [30], Hindi–English (H–E) and Marathi–English (M–E)	Orthographic		Recall H–E	29.52% 21.63%

Table 4. Performance of transliterated search techniques.

Team name, year	Model	*MAP*	*MRR*
NTNU-3, 2013	Lucene IR, TF-IDF	0.197	0.593
GU-1, 2013	Language model (LM), TF-IDF (Roman, Devanagari)	0.255	0.584
TUVal-2, 2013	DFR (2*, 0.95**, XSrqA_M***)	0.424	0.844
BIT-2, 2014	DFR	0.3415	0.6271
BITS-Lipyantran-2, 2014	Sub-word indexing and blind relevance feedback	0.6421	0.8171
DCU-1, 2014	Language model (LM) query likelihood	0.4112	0.6269
IIITH, 2014	Language model (LM)	0.412	0.673

* 2 is the minimum length of terms considered for variation look-up to expand the query.

** 0.95 is threshold Theta (θ) for similarity.

*** XSrqA_M is a ranking algorithm.

- Generating techniques dealing with grapheme-based approaches performed better than their counterparts.
- Mining-based approaches outperform generating-based ones in general. Among these, phonetic-based approaches performed well.
- Hence, phonetic-based mining techniques could be opted across language pairs, with English being one of them.

8.2 *Transliteration for the Indian languages*

There could be three ways of transliteration as far as Indian languages are concerned: (i) from English-to-Indian Language (E–IL), (ii) Indian Language-to-English (IL–E) and (iii) from one Indian language to another (IL–IL). Table 5 shows the status of research for these language pairs.

Following observations can be made.

- Major research has considered E–IL pairs. Since the English is *lingua franca* in India and people mostly use English keyboards, substantial data are generated on English transliteration for ILs.
- There have been few works in the opposite direction (IL–E).
- Work involving transliteration within Indian languages is very few and in a nascent stage. Research in this area requires proper attention to enhance the accuracy of the translation and CLIR.
- In recent years, there has been huge expansion in the number of Hindi speakers across India. A large number of Indian citizens are bi-lingual or trilingual. Hence, the transliteration from Hindi to other languages needs to be a thrust area for future research.
- Many regional languages still remain un-addressed.
- Transliteration performances of E–IL systems generally are a little lower than E–X (X = Japanese, Korean, Russian, Arabic) counterparts. While less data can be one reason, more matured technology is also need of the hour.
- As the amount of transliterated text is rapidly increasing on the Web, so is the volume of search over these data. Search in transliterated domain is, therefore, a natural extension of research on transliteration globally in general, and more so in the Indian context.
- Overall, the transliteration research in Indian language needs more attention both in terms of data creation and application of techniques.

Table 5. Summary of transliteration work involving IL.

Language pairs	Generation	Mining	Fusion	Group
English–Hindi	[29, 38, 77]	[35, 98, 107]		E–IL
English–Bengali		[35, 40]		E–IL
English–Marathi				E-IL
English–Telugu		[35]		E–IL
English–Kannada	[77]	[98]		E–IL
English–Tamil		[6]		E–IL
English–Urdu		[35]		E–IL
Hindi–English	[29, 75]			IL–E
Marathi–English	[75]		[30]	IL–E
Hindi–Urdu	[39]	[35]		IL–IL

9. Conclusion

Transliteration is transformation of a term in another language by preserving its pronunciation. This transformation can happen from a native to a foreign language or vice versa. Many non-English speakers use the Roman script to write text of their native language due to a number of socio-technical issues. With the advent of Web 2.0 and different Web-enabled mobile devices, there has been a paradigm shift in the way people use the Web. From the passive consumers, more and more people have become active content contributors on the Web. A substantial amount of this user-generated content is in the transliterated domain and it is growing very rapidly. Therefore, transforming a transliterated text back to its native language/script has received huge attention in the recent past. Absence of any standard grammar, spelling guidelines coupled with pronunciation variation (regional), language-of-origin, missing sound and script specification has made the task more challenging. In this survey, we attempted to put together different facets of the problem in transliteration and recent works in the domain concerning techniques and evaluation metrics, and summarized performances of different models and methodologies. Although straightway comparison is not possible due to wide variation in language pairs, dataset used and metrics reported, *mining* approaches seem to perform relatively better than generative approaches in general. However, the mining approaches require sufficient amount of words in *parallel* or *comparable* corpora. The generation approaches are supposed to be good for transliteration; however, they suffer from spelling variations for a given term. The *fusion* approaches, undertaken only by a limited number of researchers, have so far exhibited moderate performance.

Most of these works have seen the issue of transliteration from the angle of MT. However, today not only huge content is generated in the transliterated domain but also online users are also increasingly making Web search in transliterated domain. The volume of data (both content and queries) in the transliterated domain is growing at a lightning speed. We summarized here a recent initiative on transliterated search with a small collection of Hindi song lyrics. Although the performance by different systems here is encouraging, it should be tested over a large collection. There is, therefore, an impending need to build a large collection in the Web scale in the transliterated domain. Search in the transliterated domain is a new research direction, which is slowly unfolding, but is poised to soon take the centre stage. There will be a host of issues in the indexing, search, retrieval and evaluation of this transliterated information, which the language researchers should jump upon.

References

[1] Manning C D, Raghavan P and Schütze H 2008 *Introduction to information retrieval*, vol. 1. Cambridge: Cambridge University Press

[2] Allan J, Croft B, Moffat A and Sanderson M 2012 Frontiers, challenges, and opportunities for information retrieval. *Report from SWIRL 2012: The Second Strategic Workshop on Information Retrieval in Lorne*, SIGIR Forum, vol. 46(1), pp. 2–32

[3] Qazvinian V, Rosengren E, Radev D R and Mei Q 2011 Rumor has it: identifying misinformation in microblogs. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 1589–1599

[4] Finch A, Yasuda K, Okuma H, Sumita E and Nakamura S 2011 A bayesian model of transliteration and its human evaluation when integrated into a machine translation system. *IEICE Trans. Inf. Syst.* E94-D(10): 1889–1900

[5] Zhao B, Bach N, Lane I R and Vogel S 2007 A log-linear block transliteration model based on bi-stream HMMs. In: *Proceedings of The North American Association for Computational Linguistic*, pp. 364–371

[6] El-Kahky A, Darwish K, Aldein A S, El-Wahab M A, Hefny A and Ammar W 2011 Improved transliteration mining using graph reinforcement. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 1384–1393

[7] Chang M W, Goldwasser D, Roth D and Tu Y 2009 Unsupervised constraint driven learning for transliteration discovery. In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 299–307

[8] El Kahki A, Darwish K, El Din A S and El-Wahab M A 2012 Transliteration mining using large training and test sets. In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 243–252

[9] Lam W, Huang R and Cheung P S 2004 Learning phonetic similarity for matching named entity translations and mining new translations. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pp. 289–296

[10] Li H, Min Z and Jian S 2004 A joint source-channel model for machine transliteration. In: *Proceedings of the 42nd Annual Meeting of Association for Computational Linguistics*. Association for Computational Linguistics, p. 159

[11] Meng H M, Lo W K, Chen B and Tang K 2001 Generating phonetic cognates to handle named entities in English–Chinese cross-language spoken document retrieval. In: *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU'01*. IEEE, pp. 311–314

[12] Virga P and 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*. ACM, pp. 365–366

[13] Finch A, Dixon P and Sumita E 2012 Rescoring a phrase-based machine transliteration system with recurrent neural network language models. In: *Proceedings of the 4th Named Entity Workshop*. Association for Computational Linguistics, pp. 47–51

[14] Fujii A and Ishikawa T 2001 Japanese–English cross-language information retrieval: exploration of query translation and transliteration. *Comput. Humanit.* 35(4): 389–420

[15] Goto I, Kato N, Ehara T and Tanaka H 2004 Back transliteration from Japanese to English using target English context. In: *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, p. 827

[16] Klementiev A and Roth D 2006 Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 817–824

[17] Qu Y, Grefenstette G and Evans D A 2003 Automatic transliteration for Japanese-to-English text retrieval. In: *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pp. 353–360

[18] Hong G, Kim M J, Lee D G and Rim H C 2009 A hybrid approach to English–Korean name transliteration. In: *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Association for Computational Linguistics, pp. 108–111

[19] Jong-Hoon O 2005 Machine learning based English-to-Korean transliteration using grapheme and phoneme information. *IEICE Trans. Inf. Syst.* 88(7): 1737–1748

[20] Jung S Y, Hong S and Paek E 2000 An English to Korean transliteration model of extended Markov window. In: *Proceedings of the 18th Conference on Computational Linguistics*, vol. 1. Association for Computational Linguistics, pp. 383–389

[21] Oh J H and Choi K S 2002 An English–Korean transliteration model using pronunciation and contextual rules. In: *Proceedings of the 19th International Conference on Computational Linguistics*, vol. 1. Association for Computational Linguistics, pp. 1–7

[22] Zelenko D and Aone C 2006 Discriminative methods for transliteration. In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 612–617

[23] AbdulJaleel N and Larkey L S 2003 Statistical transliteration for English–Arabic cross language information retrieval. In: *Proceedings of the twelfth International conference on Information and Knowledge Management*. ACM, pp. 139–146

[24] Al-Onaizan Y and Knight K 2002 Machine transliteration of names in Arabic text. In: *Proceedings of the ACL-02 workshop on Computational Approaches to Semitic Languages*. Association for Computational Linguistics, pp. 1–13

[25] Habash N, Soudi A and Buckwalter T 2007 On Arabic transliteration. In: *Arabic computational morphology*. Dordrecht, South Holland: Springer, pp. 15–22

[26] Karimi S, Scholer F and Turpin A 2007 Collapsed consonant and vowel models: new approaches for English–Persian transliteration and back-transliteration. In: *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, June 23–30, Prague, Czech Republic. Association for Computational Linguistics, pp. 648–655

[27] Karimi S, Turpin A and Scholer F 2006 English to Persian transliteration. In: *String processing and information retrieval*. Glasgow, UK: Springer, pp. 255–266

[28] Chinnakotla M K and Damani O P 2009 Experiences with English–Hindi, English–Tamil and English–Kannada transliteration tasks at news 2009. In: *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Association for Computational Linguistics, pp. 44–47

[29] Chinnakotla M K, Damani O P and Satoskar A 2010 Transliteration for resource-scarce languages. In: *ACM Trans. Asian Lang. Inf. Process. (TALIP), vol.* 9(4), p. 14

[30] Chinnakotla M K, Ranadive S, Damani O P and Bhattacharyya P 2008 Hindi to English and Marathi to English cross language information retrieval evaluation. In: *Advances in multilingual and multimodal information retrieval*. Budapest, Hungary: Springer, pp. 111–118

[31] Darwish K 2010 Transliteration mining with phonetic conflation and iterative training. In: *Proceedings of the 2010 Named Entities Workshop*. Association for Computational Linguistics, pp. 53–56

[32] Das A, Ekbal A, Mandal T and Bandyopadhyay S 2009 English to Hindi machine transliteration system at news 2009. In: *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Association for Computational Linguistics, pp. 80–83

[33] Das A, Saikh T, Mondal T, Ekbal A and Bandyopadhyay S 2010 English to Indian languages machine transliteration system at news 2010. In: *Proceedings of the 2010 Named Entities Workshop*. Association for Computational Linguistics, pp. 71–75

[34] Durrani N, Hoang H, Koehn P and Sajjad H 2014 Integrating an unsupervised transliteration model into statistical machine translation. In: *Proceedings of the 15th Conference of the European Association for Computational Linguistics*. Association for Computational Linguistics, pp. 148–154

[35] Durrani N and Koehn P 2014 Improving machine translation via triangulation and transliteration. In: *Proceedings of the 17th Annual Conference of the European Association for Machine Translation*, pp. 9–17

[36] Durrani N, Sajjad H, Fraser A and Schmid H 2010 Hindi-to-Urdu machine translation through transliteration. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 465–474

[37] Gupta K, Choudhury M and Bali K 2012 Mining Hindi–English transliteration pairs from online Hindi lyrics. In: *Proceedings of the LREC*, pp. 2459–2465

[38] Gupta P, Bali K, Banchs R E, Choudhury M and Rosso P 2014 Query expansion for mixed-script information retrieval. In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, pp. 677–686

[39] Malik M G, Boitet C and Bhattacharyya P 2008 Hindi–Urdu machine transliteration using finite-state transducers. In: *Proceedings of the 22nd International Conference on Computational Linguistics*. Association for Computational Linguistics, vol. 1, pp. 537–544

[40] Dasgupta T, Sinha M and Basu A 2013 A joint source channel model for the English to Bengali back transliteration. In: *Mining intelligence and knowledge exploration*. Springer, pp. 751–760

[41] Ekbal A, Naskar S K and Bandyopadhyay S 2006 A modified joint source-channel model for transliteration. In: *Proceedings of the COLING/ACL Main conference poster sessions*. Association for Computational Linguistics, pp. 191–198

[42] Ekbal A, Naskar S K and Bandyopadhyay S 2007 Named entity recognition and transliteration in Bengali. *Lingvist. Investig.* 30(1): 95–114

[43] Malik M G 2006 Punjabi machine transliteration. In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 1137–1144

[44] Rama T and Gali K 2009 Modeling machine transliteration as a phrase based statistical machine translation problem. In: *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Association for Computational Linguistics, pp. 124–127

[45] Antony P J, Ajith V P and Soman K P 2010 Kernel method for English to Kannada transliteration. In: *Proceedings of ITC '10: Proceedings of the 2010 International Conference on Recent Trends in Information, Telecommunication and Computing*. IEEE, pp. 336–338

[46] Antony P J, Ajith V P and Soman K P 2010 Statistical method for English to Kannada transliteration. In: *Proceedings of the International Conference on Recent Trends in Business Information Processing—BAIP 2010*. Trivandrum, India: Springer, pp. 356–362

[47] Janarthanam S C, Subramaniam S and Nallasamy U 2008 Named entity transliteration for cross-language information retrieval using compressed word format mapping algorithm. In: *Proceedings of the 2nd ACM workshop on Improving non-English Web Searching*. ACM, pp. 33–38

[48] Karimi S, Scholer F and Turpin A 2011 Machine transliteration survey. *ACM Comput. Surv.* 43(3): 1–46 (article 17)

[49] Antony P and Soman K 2011 Machine transliteration for Indian languages: a literature survey. *Int. J. Sci. Eng. Res.* 2: 1–8

[50] Chen C H and Hsu C C 2008 Boosted voting for confirming synonymous transliteration. In: *Proceedings of the International Conference on Information and Automation, 2008* ICIA 2008. IEEE, pp. 1337–1342

[51] Hermjakob U, Knight K and Daumé III H 2008 Name translation in statistical machine translation—learning when to transliterate. In: *Proceedings of the ACL*, pp. 389–397

[52] Kawtrakul A, Deemagarn A, Thumkanon C, Khantonthong N and McFetridge P 1998 Backward transliteration for Thai document retrieval. In: *Proceedings of The 1998 IEEE Asia–Pacific Conference on Circuits and Systems,* Chiangmai. IEEE

[53] Karimi S, Turpin A and Scholer F 2007 Corpus effects on the evaluation of automated transliteration systems. In: *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, June 23–30, Prague, Czech Republic. Association for Computational Linguistics, pp. 640–647

[54] Sherif T and Kondrak G 2007 Substring-based transliteration. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL), June 23–30, Prague, Czech Republic. Association for Computational Linguistics, pp. 944–951

[55] Stalls B G and Knight K 1998 Translating names and technical terms in Arabic text. In: *Proceedings of the Workshop on Computational Approaches to Semitic Languages*. Association for Computational Linguistics, pp. 34–41

[56] Jeong K S, Myaeng S H, Leeb J S and Choib K S 1999 Automatic identification and back-transliteration of foreign words for information retrieval. *Inf. Process. Manag.* 35: 523–540

[57] Kang I H and Kim G 2000 English-to-Korean transliteration using multiple unbounded overlapping phoneme chunks. In: *Proceedings of the 18th Conference on Computational Linguistics*, vol. 1. Association for Computational Linguistics, pp. 418–424

[58] Gao W, Wong K F and Lam W 2005 Improving transliteration with precise alignment of phoneme chunks and using contextual features. In: *Information retrieval technology*. Beijing, China: Springer, pp. 106–117

[59] Li H, Kuo J S, Su J and Lin C L 2008 Mining live transliterations using incremental learning algorithms. *Int J. Comput. Process. Lang.* 21(02): 183–203

[60] Li H, Sim K C, Kuo J S and Dong M 2007 Semantic transliteration of personal names. In: *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, June 23–30, Prague, Czech Republic. Association for Computational Linguistics, pp. 120–127

[61] Min Z, Li H and Jian S 2004 Direct orthographical mapping for machine transliteration. In: *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, p. 716

[62] Wan S and Verspoor C M 1998 Automatic English–Chinese name transliteration for development of multilingual resources. In: *Proceedings of the 17th International Conference on Computational Linguistics*, vol. 2. Association for Computational Linguistics, pp. 1352–1356

[63] Xu L, Fujii A and Ishikawa T 2006 Modeling impression in probabilistic transliteration into Chinese. In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 242–249

[64] Aramaki E, Imai T, Miyo K and Ohe K 2006 Support vector machine based orthographic disambiguation. In: *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation*, Citeseer, pp. 21–30

[65] Aramaki E, Imai T, Miyo K and Ohe K 2008 Orthographic disambiguation incorporating transliterated probability. In: *Proceedings of the International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, pp. 48–55

[66] Bilac S and Tanaka H 2005 Direct combination of spelling and pronunciation information for robust back-transliteration. In: *Computational linguistics and intelligent text processing*. Springer, pp. 413–424

[67] Knight K and Graehl J 1998 Machine transliteration. *Comput. Linguist.* 24(4): 599–612

[68] Oh J H and Isahara H 2007 Machine transliteration using multiple transliteration engines and hypothesis re-ranking. In: *Proceedings of MT Summit XI*, pp. 353–360

[69] Lindén K 2006 Multilingual modeling of cross-lingual spelling variants. *Inf. Retr.* 9(3): 295–310

[70] Loponen A, Pirkola A, Järvelin K and Keskustalo H 2008 A novel implementation of the FITE-TRT translation method. In: *Proceedings of the 30th European Conference on Advances in Information Retrieval*. Springer-Verlag, pp. 138–149

[71] Pirkola A, Toivonen J, Keskustalo H and Järvelin K 2006 FITE-TRT: a high quality translation technique for OOV words. In: *Proceedings of the 2006 ACM Symposium on Applied Computing*. ACM, pp. 1043–1049

[72] Toivonen J, Pirkola A, Keskustalo H, Visala K and Järvelin K 2005 Translating cross-lingual spelling variants using transformation rules. *Inf. Process. Manag.* 41(4): 859–872

[73] Knight K and Graehl J 1997 Machine transliteration. In: *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics

[74] Ravi S and Knight K 2009 Learning phoneme mappings for transliteration without parallel data. In: *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 37–45

[75] Dhore M L, Dhore S K and Dixit R M 2012 Optimizing transliteration for Hindi/Marathi to English using only two weights. In: *Proceedings of the 24th International Conference on Computational Linguistics*, Bombay, India, pp. 31–48

[76] Zhang M, Duan X, Pervouchine V and Li H 2010 Machine transliteration: leveraging on third languages. In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, pp. 1444–1452

[77] Kumaran A, Khapra M M and Bhattacharyya P 2010 Compositional machine transliteration. *ACM Trans. Asian Lang. Inf. Process.* 9(4): 1–28

[78] Wang Y C, Wu C K and Tsa R T H 2015 NCU IISR English–Korean and English–Chinese named entity transliteration using different grapheme segmentation approaches. In: *Proceedings of NEWS 2015: The Fifth Named Entities Workshop*, p. 83

[79] Oh J H and Choi K S 2005 An ensemble of grapheme and phoneme for machine transliteration. In: *Proceedings of Natural Language Processing–IJCNLP 2005*. Jeju Island, South Korea: Springer, pp. 450–461

[80] Oh J H and Choi K S 2006 An ensemble of transliteration models for information retrieval. *Inf. Process. Manag.* 42(4): 980–1002

[81] Bilac S and Tanaka H 2004 A hybrid back-transliteration system for Japanese. In: *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, p. 597

[82] Yao L and Kondrak G 2015 Joint generation of transliterations from multiple representations. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pp. 943–952. http://www.aclweb.org/anthology/N15-1095

[83] Wang D, Yang X, Xu J, Chen Y, Wang N, Liu B, Yang J and Zhang Y 2015 A hybrid transliteration model for Chinese–English named entities—BJTU-NLP report for the 5th Named Entities Workshop. In: *Proceedings of NEWS 2015: The Fifth Named Entities Workshop*, vol. 67

[84] Huang F 2005 Cluster-specific named entity transliteration. In: *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 435–442

[85] Hagiwara M and Sekine S 2012 Latent semantic transliteration using dirichlet mixture. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Jeju, Republic of Korea, July 8–14, 2012, pp. 30–37

[86] Hagiwara M and Sekine S 2011 Latent class transliteration based on source language origin. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, Portland, Oregon, June 19–24, 2011, pp. 53–57

[87] Josan G S and Lehal G S 2010 A Punjabi to Hindi machine transliteration system. *Comput. Linguist. Chin. Lang. Process.* 15(2): 77–102

[88] Josan G S and Kaur J 2011 Punjabi to Hindi statistical machine transliteration. *Int. J. Inf. Technol. Knowl. Manag.* 4(2): 459–463

[89] Khapra M M, Ramanathan A, Kunchukuttan A, Visweswariah K and Bhattacharyya P 2014 When transliteration met crowdsourcing: an empirical study of transliteration via crowdsourcing using efficient, non-redundant and fair quality control. In: *Proceedings of the 9th Language Resources and Evaluation Conference (LREC)*

[90] Kondrak G and Sherif T 2006 Evaluation of several phonetic similarity algorithms on the task of cognate identification. In: *Proceedings of the Workshop on Linguistic Distances*. Association for Computational Linguistics, pp. 43–50

[91] Kuo J S and Yang Y K 2004 Constructing transliteration lexicons from web corpora. In: *Proceedings of ACL 2004 on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, p. 3

[92] Kuo J S and Yang Y K 2005 Incorporating pronunciation variation into extraction of transliterated-term pairs from web corpora. *J. Chin. Lang. Comput.* 15(1): 33–44

[93] Kuo J S, Li H and Yang Y K 2007 A phonetic similarity model for automatic extraction of transliteration pairs. *ACM Trans. Asian Lang. Inf. Process.***6**(2) (article 6)

[94] Oh J H and Isahara H 2006 Mining the web for transliteration lexicons: joint-validation approach. In: *Proceedings of the 2006 IEEE–WIC–ACM International Conference on Web Intelligence*. IEEE Computer Society, pp. 254–261

[95] Oh J H, Choi K S and Isahara H 2006 A hybrid model for extracting transliteration equivalents from parallel corpora. In: *Text, speech and dialogue*. Brno, Czech Republic: Springer, pp. 119–126

[96] Lee C J, Chang J S and Jang J S R 2006 Extraction of transliteration pairs from parallel corpora using a statistical transliteration model. *Inf. Sci.* 176(1): 67–90

[97] Sproat R, Tao T and Zhai C 2006 Named entity transliteration with comparable corpora. In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 73–80

[98] Udupa R, Saravanan K, Kumaran A and Jagarlamudi J 2008 Mining named entity transliteration equivalents from comparable corpora. In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. ACM, pp. 1423–1424

[99] Udupa R, Saravanan K, Kumaran A and Jagarlamudi J 2009 Mint: a method for effective and scalable mining of named entity transliterations from large comparable corpora. In: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 799–807

[100] Sato S 2010 Non-productive machine transliteration. In: *Proceedings of the Conference on Adaptivity, Personalization and Fusion of Heterogeneous Information*, Le Centre De Hautes Etudes Internationales D'Informatique Documentaire, pp. 16–19

[101] Hsu C C and Chen C H 2010 Mining synonymous transliterations from the world wide web. *ACM Trans. Asian Lang. Inf. Process.* 9(1): 1–28

[102] Lee J S and Choi K S 1998 English to Korean statistical transliteration for information retrieval. *Comput. Process. Oriental Lang.* 12(1): 17–37

[103] Sherif T and Kondrak G 2007 Bootstrapping a stochastic transducer for Arabic–English transliteration extraction. In: *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, June 23–30, Prague, Czech Republic. Association for Computational Linguistic, pp. 864–871

[104] Goldwasser D and Roth D 2008 Active sample selection for named entity transliteration. In: *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*. Association for Computational Linguistics, pp. 53–56

[105] Kuo J S, Li H and Yang Y K 2008 Active learning for constructing transliteration lexicons from the web. *J. Am. Soc. Inf. Sci. Technol.* 59(1): 126–135

[106] Kaur J and Josan G S 2011 Statistical approach to transliteration, from English to Punjabi. *Int. J. Comput. Sci. Eng.* 3(4): 1518–1527

[107] Fukunishi T, Finch A, Yamamoto S and Sumita E 2013 A Bayesian alignment approach to transliteration mining. *ACM Trans. Asian Lang. Inf. Process.* 12(3): 9

[108] Sajjad H, Fraser A and Schmid H 2012 A statistical model for unsupervised and semi-supervised transliteration mining. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers*, vol. 1. Association for Computational Linguistics, pp. 469–477

[109] Htun O, Finch A, Sumita E and Mikami Y M 2012 Improving transliteration mining by integrating expert knowledge with statistical approaches. *Int. J. Comput. Appl.* 58(17): 12–22 (full text available)

[110] Kunchukuttan A and Bhattacharyya P 2015 Data representation methods and use of mined corpora for Indian language transliteration. In: *Proceedings of NEWS 2015: The Fifth Named Entities Workshop*, p. 78

[111] Shao Y, Tiedemann J and Nivre J 2015 Boosting English–Chinese machine transliteration via high quality alignment and multilingual resources. In: *Proceedings of NEWS 2015: The Fifth Named Entities Workshop*, p. 56

[112] Finch A, Liu L, Wang X and Sumita E 2015 Neural network transduction models in transliteration generation. In: *Proceedings of NEWS 2015: The Fifth Named Entities Workshop*, p. 61

[113] Huang F and Vogel S 2002 Improved named entity translation and bilingual named entity extraction. In: *Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces*. IEEE, pp. 253–258

[114] Nagata M, Saito T and Suzuki K 2001 Using the web as a bilingual dictionary. In: *Proceedings of the Workshop on Data-driven Methods in Machine Translation*, vol. 14. Association for Computational Linguistics, pp. 1–8

[115] Tsuji K, Daille B and Kageura K 2002 Extracting French–Japanese word pairs from bilingual corpora based on transliteration rules. In: *Proceedings of LREC 2002: 3rd*

International Conference on Language Resources and Evaluation, pp. 499–502

[116] Wu C K, Wang Y C and Tsai R T H 2012 English–Korean named entity transliteration using substring alignment and re-ranking methods. In: *Proceedings of the 4th Named Entity Workshop*. Association for Computational Linguistics, pp. 57–60

[117] Qu W 2013 English–Chinese name transliteration by latent analogy. In: *Proceedings of the Fifth International Conference on Computational and Information Sciences (ICCIS)*. IEEE, pp. 575–578

[118] Narasimhulu V, Sujatha P, Dhavachelvan P and Basha M S 2010 Enhanced named entity transliteration model using machine learning algorithm. *Int. J. Adv. Comput. Technol.* 2(3): 84–93

[119] Choudhury M, Chittaranjan G, Gupta P and Das A 2014 Overview and datasets of fire 2014 track on transliterated search. In: *Pre-proceedings of the 6th FIRE-2014 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[120] Roy R S, Choudhury M, Majumder P and Agarwal K 2013 Overview of the fire 2013 track on transliterated search. In: *Proceedings of the 5th 2013 Forum on Information Retrieval Evaluation*. ACM, p. 4

[121] Pakray P and Bhaskar P 2013 Transliterated search system for Indian languages. In: *Pre-proceedings of the 5th FIRE-2013 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[122] Joshi H, Bhatt A and Patel H 2013 Transliterated search using syllabification approach. In: *Pre-proceedings of the 5th FIRE-2013 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[123] Prakash A and Saha S K 2014 A relevance feedback based approach for mixed script transliterated text search: shared task report by BIT Mesra. In: *Pre-proceedings of the 6th FIRE-2014 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[124] Bhat I A, Mujadia V, Tammewar A, Bhat R A and Shrivastava M 2014 IIIT-H system submission for fire2014 shared task on transliterated search. In: *Pre-proceedings of the 6th FIRE-2014 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[125] Mukherjee A, Datta K and Ravi A 2014 Mixed-script query labelling using supervised learning and adhoc retrieval using sub word indexing. In: *Pre-proceedings of the 6th FIRE-2014 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[126] Ganguly D, Pal S and Jones G J F 2014 DCUFIRE-2014: fuzzy queries with rule-based normalization for mixed script information retrieval. In: *Pre-proceedings of the 6th FIRE-2014 Workshop*, Forum for Information Retrieval Evaluation (FIRE)

[127] Zhang M, Li H, Banchs R E and Kumaran A 2015 Whitepaper of NEWS 2015 shared task on machine transliteration. In: *Proceedings of the 5th Named Entity Workshop*. Association for Computational Linguistics, pp. 1–9

[128] Papineni K, Roukos S, Ward T and Zhu W J 2002 BLEU 2002: a method for automatic evaluation of machine translation. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp. 311–318