Artif Intell Rev (2018) 49:339–373
https://doi.org/10.1007/s10462-016-9527-1

A survey on Urdu and Urdu like language stemmers and stemming techniques

Abdul Jabbar[1] · Sajid Iqbal[2] · Muhammad Usman Ghani Khan[3] · Shafiq Hussain[4]

Published online: 28 November 2016
© Springer Science+Business Media Dordrecht 2016

Abstract Stemming is one of the basic steps in natural language processing applications such as information retrieval, parts of speech tagging, syntactic parsing and machine translation, etc. It is a morphological process that intends to convert the inflected forms of a word into its root form. Urdu is a morphologically rich language, emerged from different languages, that includes prefix, suffix, infix, co-suffix and circumfixes in inflected and multi-gram words that need to be edited in order to convert them into their stems. This editing (insertion, deletion and substitution) makes the stemming process difficult due to language morphological richness and inclusion of words of foreign languages like Persian and Arabic. In this paper, we present a comprehensive review of different algorithms and techniques of stemming Urdu text and also considering the syntax, morphological similarity and other common features and stemming approaches used in Urdu like languages, i.e. Arabic and Persian analyzed, extract main features, merits and shortcomings of the used stemming approaches. In this paper, we also discuss stemming errors, basic difference between stemming and lemmatization and coin a metric for classification of stemming algorithms. In the final phase, we have presented the future work directions.

✉ Sajid Iqbal
Sajid.iqbal@bzu.edu.pk

Abdul Jabbar
a.jabbar73@hotmail.com

Muhammad Usman Ghani Khan
usman.ghani@kics.edu.pk

Shafiq Hussain
shafiqhussain@bzu.edu.pk

1 Department of Computer Science, Institute of Southern Punjab, Multan, Pakistan

2 Department of Computer Science, Bahauddin Zakariya University, Multan, Pakistan

3 Al-Khwarzmi Institute of Computer Science, University of Engineering and Technology, Lahore, Pakistan

4 Department of Computer Science, Bahauddin Zakariya University (Sahiwal Sub-campus), Multan, Pakistan

Keywords Stemming · Natural Language Processing · Information Retrieval · Urdu · Suffixes · Stemming Techniques

1 Introduction

In the previous two decades, a huge growth in multi-lingual, multi-mode data generation has been observed (Cambria and White 2014). Like data in other languages, textual data of Urdu language has found its own way in cyber world (Abbas 2012). Being one of the major languages of Indian sub-continent, the national language of Pakistan and one of the national languages of India, a number of websites, blogs, entertainment, digital literary resources and computing standards (Hussain and Afzal 2001) have been developed. This has automatically opened new vistas of Urdu language communication and processing for various purposes like Information Retrieval (IR), Machine Translation (MT), Natural Language Processing (NLP) and many more application areas. However, in fact primary resources required for Urdu language processing are still sparse and need to be developed (Hussain 2008).

A stemming algorithm is a computational procedure that reduces all the words with same root or stem by stripping each word's derivational and inflectional suffixes (Lovins 1968). Chris (1990), describes that to get the stem/root of the word, remove its affixes, if this could be done correctly, all the variant forms of a word would be converted to the same standard form. Khoja and Garside (1999) have defined the stemming algorithm in Arabic language context. Khoja and Garside (1999) have coined the stemming process as "Stemming is the process of removing all of a word's prefixes, suffixes and infixes to produce the stem or root". A stemmer is one of the basic language processing tools with its main purposes to cluster the words according to stem. For each application the process may be used for different purpose. In IR stemming helps to reduce the number of documents retrieved for language processing (Carpineto and Romano 2012). In NLP, stemming helps to reduce the number of words (features) for text analysis (Piotrowski 2012) and in MT, it allows to choose a correct foreign language translation for a given word considering the context (Madnani et al. 2012).

1.1 Stemming and lemmatizing

Both have the same basic function, however, stemming produces 'stem' while lemmatization returns 'lemma'. Lemmatization retrieves dictionary word or context of the word in the given sentence which must be a valid word of the language. On the other hand, a stemmer returns the common portion of the variant word forms without having knowledge of the context and the stem may be an invalid word. A stemmer may return "retriev" for words retrieving retrieved and retrieval whereas lemmatizer will return "retrieve" a correct and meaningful dictionary word (Korenius et al. 2004). There are applications where lemmatization is useful, but steaming is not. For example, lemmatization may also help in synonym search by using thesaurus whereas stemming cannot do this (Balakrishnan and Lloyd-Yemoh 2014).

1.2 Errors in stemming

Stemming errors are the erroneous results produced by a stemming algorithm. Stemming errors are classified into three groups: under-stemming, over-stemming and miss-stemming. In over stemming, two words with different stems are stemmed to the same root. In under stemming, the stemmer strips the given word below expected level. Both errors decrease the performance of the stemmer, but in different ways. To avoid over-stemming, set a minimum

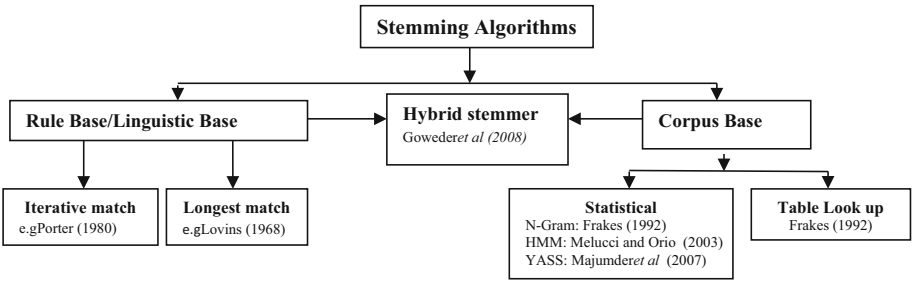

Fig. 1 Classification of stemming algorithms

size of the constraint of the derived stem/root. Under stemming can be avoided by techniques like partial-match algorithms. Stemmers may ignore some words and do not perform their stemming at all. Such words are grouped as miss-stemmed words (Al-Shammari 2013).

1.3 Classification of stemming techniques

Based on errors, stemmers are classified into two main classes: strong (or heavy) stemmers and weak (or light) stemmers. A strong stemmer has a high rate of over-stemming whereas weak steamer has a high rate of under-stemming. We define the stemmer with high miss-stemming rate as a feature called discount stemming. Researchers have used different approaches for better and efficient stemming. Lovins (1968) describes two types of stemming algorithm on the basis of principles that used in the construction of a stemming: iteration and longest-match. Hadni et al. (2012) has given three categories of Arabic language stemmers i.e. root based stemmers, statistical stemmers and light stemmers with the additional hybrid stemmer approach of their own. Root based approaches use morphological analysis to extract the root of the given word. Stem based approaches, on the other hand, do not produce the root of the word, but try to remove the suffixes from the word. Using some similarity measure, statistical methods have been experimented with some stemming methods. On the bases of approaches used for stemming process, stemming algorithms can be divided into two main categories and sub categories as shown in Fig. 1.

2 Stemming techniques used in Urdu and Urdu like languages

Urdu and Urdu like languages, i.e. Persian and Arabic are morphologically rich languages. The degree of morphological richness is higher in Arabic language (Habash 2007) than Urdu and Persian (El Kholy et al. 2013) whereas Urdu stands at last. Urdu language written in Arabic script contains lots of words from Arabic (Tahir 2014) and Persian (Khansir and Mozafari 2014) due to Arabic and Persian speaking Muslim rule in Indian sub-continent.

2.1 Stemming in Urdu language

In Urdu language, words either uni-gram or bi-gram, may be present in inflectional forms that contain added suffixes i.e. prefix, postfix and infix, along with stem word. There are inflectional forms that even reduce or change characters in stem word and additionally may use suffixes. Different researchers, who worked on stemming issue in Urdu language, have focused either one type of suffix or two types of suffixes but neither on all three simulta-

Table 1 Example add character list (ACL)

Add Character	Words
ت	قیاد،کاش،شد
ہ	آزمود،افتاد
ی	ترق
ا	انتہ،ایذ،کت

neously. Following section reviews the stemmers of Urdu language published by research community.

Akram et al. (2009) proposed an algorithm for first Urdu stemmer. It works into three phases as follows: The system maintains four lists i.e. Prefix Global Exception List (PrGEL), Prefix Rule Exception List (PrREL), Postfix Global Exception List (PrGEL) and Add Character List (ACL). In the first phase, if a given word possess a prefix and this prefix is found neither in PrGEL nor in PrREL then remove that prefix and go to next phase. For example, for the query word با اخلاق, Stemmer identifies and chops up the prefix to derive the stem اخلاق. If word does not possess any prefix, then it returns original word and proceeds to the next phase that is suffix removal. If a word contains suffix and this suffix is found neither in Postfix Global Exception Lists (PrGEL) nor in Postfix Global Exception Lists (PrGEL), then truncates the suffix and next phase is started. If given word does not contain any suffix, then it returns original word and the algorithm proceeds to the next phase. For example, for Urdu word زندگی this stemmer identifies the suffix گی and chops up the suffix and derives word زند and proceeds to the next step. In the last phase, after the prefix/postfix extraction, obtained stem is looked up in the Add Character Lists (ACL), if found, its corresponding letter is appended at the end of the word and returned as a stem. Examples of Add Character List (ACL) are given in Table 1. If word is not found in the list, then is returned as a stem. For example, زندگی سے زندہ in the example the word زند got from prior step and addہ at the end of زند.

Khan et al. (2012) proposed Light Weight Stemmer (LWS) for Urdu Language. Like Akran, Qurant-ul-ain (2009), LWS also maintains lists of affixes and affix exceptions, and operates in four steps. The lists maintained include Stop Words List (SWL), Global Prefix Exception List (GPEL) and Global Suffix Exception List (GSEL). In first step query word is checked in SWL, if found, ignore the query word and move to the next word. If query word is not found in the stop word list then go to the next step. In the second step, prefix of query word is checked in GPEL, if found, ignore the prefix and move to the next step, otherwise remove the prefix according to prefix list in descending order on longest match basis, and proceed to next step. For example, the query word is بااخلاق, Stemmer identifies and chops up the prefix and derives the stem اخلاق. In third step, the suffix is checked in Global Suffix Exception List (GSEL), if found, Stemmer returns it as stem otherwise truncates the suffix with respect to the suffix list and goes to next step. For example, Urdu word is زندگی this stemmer identifies the suffix گی and chops up the suffix to derive word زند and proceed the next step. In the fourth and final step, the query word is checked in add character list (Table 1), if found, then its corresponding character is appended at the end of derived word otherwise returns as a stem. For example, زندگی سے زندہ in the example the word زند got from prior step and add ہ at the end of زند to produce the stem زندہ.

Lehal (2012) proposed another rule base stemmer, which is different from previous stemmers. They maintain a database of 101,483 root words along with their frequencies to extract the stem. Lehal (2012) proposed stemmer that firstly tokenizes the query word, using space as delimiter and normalized special characters. Next, postfix/prefix rules are applied to the query word to generate the list of various possible stem words. After that, a possible stem word with the highest frequency is derived as a stem. If relevant rules are not found in rule base then original word is returned.

Stemmer presented by Gupta et al. (2013) works in similar way as that presented by Akram et al. (2009) stemmer but they do not include the Prefix Global Exception Lists (PrGEL),Postfix Global Exception Lists (PoGEL), Prefix Rule Exception List (PrREL),Postfix Rule Exception List (PoREL) and Add Character List (ACL). They use only prefix and suffix lists to get the stem of the given word. In this algorithm a given word is checked in the prefix and/or suffix lists, if prefix/suffix or both found, then remove the appropriate affixes and the stem is derived.

Husain et al. (2013) presented statistical stemmer for Urdu and Marathi languages. Entered word is split into n-gram e.g.

بےادب (Misbehaved) ={(Null,بےادب),(ب,ےادب),(بے,ادب),(بےا,دب),(بےاد,ب),(Null,بےادب)} and stem and suffix lists are generated. Derived suffixes are arranged in decreasing order of their frequency. Two methods are used to delete the suffixes: frequency based and suffix length based. Results of both methods are compared and frequency base approach is found better than length base method. Husain et al. (2013) test their stemmer on Marthi and Urdu languages. For Urdu language, stemming results for frequency based approach are 84.27% as compared to 79.63% of length based approach.

Mubashir Ali and Saleemi (2014) stemmer is similar to Akram et al. (2009) stemmer except they do not include Prefix Rule Exception List (PrREL) and Postfix Rule Exception List (PoREL). This stemmer operates in four steps: firstly, query word is checked in the Stop Word List (SWL), if found it is ignored and the next word is taken as query word and process is started again. Secondly, if given word contains prefix and this prefix is not present in Prefix Global Exception List (PrGEL) then apply the prefix rules and proceed to the next step. If prefix is not found, then the word is marked as processed word and the next phase is initiated. Thirdly, if given word contains prefix and this prefix is not found in Prefix Global Exception List (PrGEL) then apply the suffix rules and proceed to the next step. Finally, stem extracted in prior step, is checked in Add Character Lists (ACL) (Table 1), if found add the appropriate letter at the end of the word and return as stem. If word is not found then mark the word as processed word and return as stem.

Khan et al. (2015) enhanced the (Akram et al. 2009) stemmer. They described their stemmer by using prefix and suffix lists, Global Prefix Exception List (GPrEL), Suffix Exception List (GSEL), Stop Words List (SWL), Add Character List (ACL), Words Pattern List (WPL) and Stem Words Dictionary (SWD). For Urdu words of length 4, two patterns are given "فعال" and "فعول" if third character is "ا" or "و" then remove it. For example, حکام سے حکم، بروج سے برج For Urdu words of length five characters three patterns are given "فعالا", "مفاعل" and "افاعل"; and for Urdu words of length six characters one pattern is given "فعالیل" and on that pattern words, stem is derived. For the words having Hamza at the end and have length four characters, then check if first character is "ا" and the last character is "ء" then add the third character "ی" and remove the last "ء". Khan et al. (2015) extracted the stem, using six steps. First, query word is checked in the stop word list, if found, return the original word otherwise proceed the next step. Second, if the query word possesses prefix and this prefix is not found in Global Prefix Exception List (GPrEL), then remove that prefix

and proceed. If a word has no prefix, then pass the word unchanged to the next step. Third, if a word has suffix and this suffix is not found in Global Suffix Exception List (GSEL), then remove the suffix and proceed. If the word does not contain any suffix, then the unchanged word is passed to the next step. In fourth step, after removing the prefix and suffix, extracted stem word is checked in Add Character Lists (ACL), if found, then its corresponding letter is added at the end of the word and proceed the next step. If word is not present in Add Character Lists (ACL), then the unchanged word is passed to the next step. In fifth step, derived stem is checked into the stem words dictionary, if found, it is returned as a stem otherwise it contains infixes and proceeds to next step. Finally, infixes are removed according to given words patterns to extract the stem.

Gupta et al. (2015) improved their prior work (Gupta et al. 2013). This stemmer firstly checks whether the input is sentence or a word, if input is a word, then this word is looked up into the database containing stop words and exception words (solid stem), if found it returns the original word otherwise apply the affix rules and extract the stem. If the input is sentence, it is first tokenized and then each token is processed as a single word.

Discussion

The deficiency of Akram et al. (2009) stemmer is that it only removes the prefixes and suffixes, along with this, it has some exception words e.g. ترکیبات،شامیانےنگاہیں وغیرہ and has also exception lists of suffixes and prefixes. Lehal (2012) approach does not handle compound words that contain hard space because they delimit the words by a space, e.g. خوش مزاج،چراغ دان Further, its correctness depends upon the database, if the stem of query word is not present in the database then it will return unstem or wrong stem. Gupta et al. (2013) do not define any mechanism to handle solid stem i.e. some suffixes are actually part of the word and need not to be removed e.g. ٹوکری, this case is not handled in this work. Khan et al. (2012) does not deal with infixes. Husain et al. (2013) proposed method is not suitable for handling broken plural words and infixes. Mubashir Ali and Saleemi (2014) ignore the compound words and do not deal with infixes. The limitations of Khan et al. (2015) stemmer are: first, they proposed only two methods for Urdu words length four (4) character but these two method have their own exceptions that are not included e.g. وحوش سے وحشی moreover many Urdu words of length four are not handled e.g. اطبا سے طبیب، کبیر سے اکبر،افاق سے افق وغیرہ Second, for five character Urdu words, four methods are proposed and still many words of having five characters are not stemmed e.g. اسماء سے اسم ، امراء سے امیر، مکاتب سے مکتب اور ثوابت سے ثابت وغیرہ Third, for six character Urdu words, only one method is proposed and many words having six characters are not handled e.g. تواریخ سے تاریخ، تصاویر سے تصویر، انبیاء سے نبی وغیرہ Fourth, Infix handling rules are incomplete as in pattern "فعال" only third character is removed to get the stem but in many cases, only deletion is not sufficient to get the stem e.g. تراجم سے ترجمہ،سلاسل سے سلسلہ Fourth, every pattern has an exception list that is not included e.g. pattern "فعول" has exception words جلوس سے جلسہ ، حصول سے حاصل اور Fifth, pattern list is incomplete e.g. فعیل،فاعل،افتعال،تفعیل Sixth, many words having infix are not handled by any pattern e.g. the word "میت" and its root is "موت". Gupta et al. (2015) proposed stemmer does not remove the stop words that produce a linguistically incorrect sentence, for instance ہم پاکستانی ہیں سے ہم پاکستان ہیں. Moreover, criteria for sentence tokenization are not mentioned.

Most of Urdu stemmers, presented till now, use exception lists of words/or rules and some add stem word dictionary too. The performance of these stemmers is fully dependent on the lists, if a stemable word does not match with exception rules or in list, it remains unstemmed that reduces the performance. On the other side, these stemmers are evaluated by manual methods, i.e. human expert evaluation and a few researchers have used quantitative perfor-

mance measures like recall, precisian and F-measure to evaluate their stemmers. Another issue is the use of small data set for evaluation that leads to biased results. There are few quality data sets available and none of them is standardized. Further, no comparison is made to show the improvement in performance from prior presented stemmers because every stemmer uses different type and size of lists and corpus. Moreover, it is not possible to verify their claims of accuracy because source codes and the data sets used in the testing process are not available openly. In Table 2, we compare the different parameters of reviewed Urdu stemmers.

2.2 Stemming in Arabic language

Arabic is one of the major languages that has got the attention of Natural Language Processing (NLP) community, especially in the US and is mostly researched language among Persian and Urdu. The language is also considered as one of the toughest languages for NLP (Zughoul et al. 2005; Ebrahim et al. 2015). A good number of stemmers, for Arabic, have been designed and published by researchers. As the review shows, powerful stemming approaches have been formulated and implemented in different NLP applications (Al-Kabi et al. 2013; Al-Shammari 2013; Dahab and Al-Mutawa 2015).

Superior root-based stemmer, a rule based Arabic stemmer, presented by Khoja and Garside (1999), works in different phases. In first phase, the stemmer removes the longest suffix and the longest prefix. In the next step, it matches the remaining word with the verb and noun patterns, to take out the root. At TREC 2002 (Chen and Gey 2002), developed two Arabic stemmers, an MT-based stemmer and a light stemmer. Machine translation (MT) based stemmer employs a unique approach. It clusters the Arabic words based on their English translations and the shortest word in a cluster are considered as the stem word. This stemmer transforms the broken plural forms of an Arabic word into its singular form. Light stemmer uses predefined lists of prefixes and suffixes. It removes the prefixes non-recursively and removes the suffixes recursively to get the stem word.

Nwesri et al. (2005) presented stemming for Arabic conjunctions and prepositions based on grammatical rules. They introduced four rules using Arabic grammar, based on these four rules, they presented three new algorithms: Remove and Check in Lexicon (RCL); Replace and Remove (RR); and Replicate and Remove (RPR). First algorithm, Remove and Check in Lexicon (RCL) checks the first letter of the word, if first letter belongs to given sets, then it is removed and remaining part is checked in the dictionary. If remaining part is found, it is returned as root, otherwise original word is output. Second algorithm, Replace and Remove (RR) checks the possible particles by replacing the first letter of the given word with *waw* (و) and *faa* (ف), and checks in the dictionary. If both words are found in the dictionary, it shows that the first letter of the original word is a particle, therefore, removes it and extracts the root word. For example, from the Arabic word وبطاقتي (my id card) the stem طاقتي (my power) is obtained by Replace and Remove (RR) stemmer. Third algorithm, Replicate and Remove (RPR) checks whether a given word is verbal or not by first removing first letter and then duplicating first letter. Both new words cut-down versions of the word and extended version of word are checked in the dictionary, if either word is found, original word is returned without any stemming. For example, from the Arabic word وبطاقتي (my id card) the stem بطاقتي (my power) by Replicate and Remove (RPR).

Taghva et al. (2005b) proposed Arabic stemmer without a root dictionary. They created one to three letter suffix and prefix lists, and word patterns. Examples of Arabic word patterns are given in Table 3.

Using these word patterns prefixes and suffixes are removed iteratively until stem words length is not less than three. It is similar to the Khoja's stemmer (Khoja and Garside 1999) in

Table 2 Comparison of Urdu stemming technique

Authors	Stemmers	Used Approach	Required lists	Benefits	Limitations
Akram et al. (2009)	Assas-band, an affix-exception-list based Urdu stemmer	Rule base	Prefix and suffix lists	Deleted the prefixes and suffixes where ever applicable	Slow execution because a lot of lists are used
			Prefix global exceptional lists (PrGEL)	Stem dictionary is not used	Large space required to store the lists
			Postfix global exceptional lists (PoGEL)	May be used in information retrieve (IR), natural language processing (NLP) and text mining applications	Not deal with infixes
			Postfix rule exceptional list (PoREL)	Over stemming may occur due to longest affixes matching remove first	Compound words are not handled
			Prefix rule exceptional list (PrREL)		Not deal with irregular plural words
			Add character list (ACL)		Suffix and prefix list is incomplete
Khan et al. (2012)	A light weight stemmer for Urdu language	Rule base	Prefix and Suffix Lists	This stemmer removes only prefixes and suffixes wherever applicable	Slow execution because lots of lists are used
			Global prefix exceptional list (GPEL)	This type of stemmer give better result in information retrieve (IR) field	High storage space required
			Global suffix exceptional list (GSEL)		Not deal with infixes
			Add character list (ACL)		Compound words are not handled
			Stem dictionary		Not deal with irregular plural words
					Suffixes and prefix list is incomplete

Table 2 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Lehal (2012)	Rule based Urdu stemmer	Rule base	Database of 101,483 unique words	Remove the prefix and suffix attached with single words	Storage required for stem words along with frequency
				Prefix and suffix list are not use	Time consuming because all possible stem words derived
				Moreover exception words and exceptions rules are not include	Compound words are not handled
				Good for information retrieval application	Infix does not handle
Gupta et al. (2013)	Rule based stemmer in Urdu	Rule base	Prefixes and suffixes list	Remove appropriate prefixes and suffixes	Not deal with infixes
				Fast in execution as less lists are used	Small data set is used for testing
				Good for information retrieval application	Not deal with infixes
				Simple to implement	Compound words are not handled
					Not deal with irregular plural words
Husain et al. (2013)	An unsupervised approach to develop stemmer	Statistical	Instead of lists corpus used	Language Independent stemmer	Large space required to store corpus
				Good for information retrieve applications	Broken plural words are not handles
				Dictionary less stemmer	Slow execution
				Better for text mining	Compound words are not handled
				Based on concept of n-grams	Not applicable for infixes

Table 2 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Mubashir Ali and Saleemi (2014)	A novel stemming approach for Urdu language	Rule base	Stop word list	Include the stop words that improve the stem	High storage space required
			Prefix rules list	Low error rate	Not handle irregular plural words
			Suffix rules list	Suitable for information retrieve applications	Not remove infixes
			Prefix global exception list (PrGEL)	Use in text mining application	Committed under stemming and over stemming
			Postfix global exception list (PoGEL)	Only remove the prefixes and suffixes wherever applicable	Compound words are not handled
			Add character lists (ACLs)	Fast in execution	
			Stem word dictionary		
Khan et al. (2015)	Template based affix stemmer	Rule base and template matching	Prefix and suffix lists	This type of stemmer committed low error rate	The space required to store rules, patterns and lists
			Global prefix exceptional list (GPEL)	Include the stop words that improve the stem	Not handle irregular plural words
			Suffix exceptional list (GSEL) Stop words list	Reduce vocabulary size Good for information retrieve	Time consuming because every word has to pass through one stem word dictionary and pattern matching also
					Incomplete infixes handling rules list
			Add character list (ACL)	Remove prefixes, suffixes and some infixes	Committed under stemming and over stemming
			Words pattern list		Compound words are not handle
			Stem words dictionary		Template list incomplete

Table 2 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Gupta et al. (2015)	Urdu stemmer 'Usal'	Rule base	Prefix and suffix lists	Handle solid stem	Space required storing prefixes, suffixes, stop words and exception words
			Exception words and stop words lists	Remove prefixes and suffixes wherever applicable	Produce linguistically incorrect sentence
				Handle the sentences and extract the stem able words, and derived the stem	Criteria for sentence tokenization are not mention
				Good for information retrieve	Slow execution because every word has to pass through database
					Infixes are not consider
					Compound words are not handle
					Broken plural words are not consider
					Suffixes and prefix list is incomplete

Table 3 Arabic words pattern

Sets	Pattern
Words length four patterns	مفعل،فعيل،فعول
Words length five patterns and length three roots	تفاعل،فاعول،مفعيل
Words length five patterns and length four roots	فعالل،مفلل،تفعلل،فعلان
Words length six patterns and length three roots	مستفعل،انفعل،افتعال
Words length six patterns and length four roots	افعلال،افنلل،متفعلل

Table 4 Pattern with infix string

Pattern	Infix String
فعال	ا
مفعول	مو
يستفعلن	يستن

result but different in working. Al-Kabi and Al-Mustafa (2006) presented Arabic root based stemmer. They develop patterns up to word length nine. Examples of prefixes, suffixes and infixes that are associated with morphological pattern of words as mentioned in Table 4. If a word being processed matches one of these patterns then its corresponding prefixes and/or suffixes and/or infixes are removed to get stem.

Goweder et al. (2008) proposed a hybrid method of stemming by integrating three different techniques: affixes removal, dictionaries, and morphological techniques. These techniques applied individually and independently to solve associated stemming problems. In affix removal technique, suffixes and prefixes lists are created and used. Based on given word's length, specified affixes are removed. The pattern of rule (right to left) is given as:

[Suffix Replacement].[Suffix].[Infix Replacement].[Infix].[Prefix Replacement].[Prefix]. [WordLength] where:

[word length]: represents the length of a word without prefix.
[prefix]: represents a prefix that could be attached to word.
[prefix Replacement]: represents a substitute of prefix.
[infix]: represents an infix that is attached to a word.
[infix Replacement]: represents a substitute of an infix.
[suffix]: represents a suffix that is attached to a word.
[suffix Replacement]: represents a substitute of a suffix.

For example, Arabic word طريقتين (two methods) has affix ين and stem طريقة (method) derived by following rule pattern:

[ة].[تين].[ ].[ ].[ ].[|بال|وال|فال|ال|كال]. [*]

A dictionary based technique has two main advantages: first handling, improper removal of some prefixes and second is to handle the Arabicized word problems. The first problem is solved by creating root words dictionary and every root words are started with (ي, ت, ف, و, ل, ب,ك , and أ) as a prefix. The second problem is solved by creating an Arabicized-word dictionary. For example, the Arabicized word كمبيوترات (computers) is found in Arabicized word dictionary and its corresponding stem كمبيوتر is extracted by chopping up

Table 5 Pattern detection and transformation rule

Rule ID	Condition	Rule	Example
R1	Length = 5 and first letter is an ا and fourth letter is an ا and the second letter is ي	Remove first and fourth letters (ا)	اشجار
R2	Length = 5 and the second letter is و and the third is an ا	Remove second letter و	جوانب
R3	Length = 4 and first letter is ا	Remove first letter (ا	اشهر

the suffix ات. In morphological analysis technique words pattern used to derive the stem of the given word. For Example, the Arabic word حيوان (animal) matches the noun pattern فعوال, letters appear as affixes are ان and stem is حيوان (animal).

Al-Shammari and Lin (2008) published the Educated Text Stemmer (ETS) that is based on stemmed documents of noun dictionary and verbs dictionary. They develop rules on the bases of syntactical structure of the words. They identify the verb and noun using three lists: NSW (Noun Stop Words): list of stop words proceeding Nouns, VSW (Verb Stop Words): list of stop words proceeding verbs and SW (Stop Words): list of stop words (including both NSW and VSW). Aggressive stemming (Khoja's root-based stemmer) performs on verb and extract root. In case of noun, light stemming is use to get the stem of given noun.

El-Beltagy and Rafea (2011) divide the Arabic stemmer into two classes: aggressive and light stemmers. An aggressive stemmer that reduces words to their roots, Khoja and Garside (1999) is an example of an aggressive stemmer, and a light stemmer produces terms that are closer to the definition of a stem for information retrieval task, e.g. Larkey light stemmer (Larkey et al. 2002). They enhance the light stemmer for Arabic text. This stemmer works in three stages: prefix removal, suffix removal and infix removal respectively. For each stage specific set of rules is defined, and each stage is independent of others. First, apply the prefix rules if found, delete that prefix and search the extracted word in stem list, if found return as stem otherwise return the original word and go to the next stage. In the second stage, suffix rules are applied, if the rule is matched, then truncate the suffix and check the validities of the derived word in stem dictionary, if found return as stem, otherwise return original word and proceed the next stage. In the third stage, broken plural words are identified by the patterns and transformed into their singular, then check the validities of the derived word in the stem word dictionary as shown in Table 5. If a word is valid, return as a stem, otherwise return the unaltered word.

Ababneh et al. (2012) developed a rule-based light stemmer. This stemmer used lists of prefixes and suffixes. Khoja's words patterns are used to check the validity of Arabic words. When a query word matches with a word pattern in the list, it is returned as it is. For example, the word "كامل" contains affix "كا" but it matches with pattern "فاعل" so affix is not deleted. If word is not matched with any pattern, then the prefixes are truncated and extracted word is checked in the pattern list. If found, then return as stem, otherwise return the word unchanged and go to the next step. In this step suffix, rules are applied, if a rule matches, then remove the suffix and match the derived word with pattern list, if match then return as stem, otherwise return original word and precede the next step. After truncating, if word's length is less than three characters, then rolls back to the original word otherwise return the word as stem, e.g. the word "ولدين" after removing the prefix and suffix the word "د" is extracted which is wrong word so roll back original word and remove only the suffix "ين" and derive the stem "ولد" which is correct. The last step is to identify the plural words and transform into its corresponding

Table 6 Singular and plural patterns with example

Plural Pattern	Singular Pattern
أفعلة/فعلاء (أطباء)	فعيل (طبيب)
فواعل/فعال/فعلاء (عقلاء)	فاعل (عاقل)
أفعال/فعلاء/أفعل (سمحاء)	فعل (سمح)

Table 7 Inflection of roots

Root	Inflection	Resulted word (previous step)
كتب	فاعل	كاتب
ركب	مفعل	مركب
غفر	استفعل	استغفر

singular word. For this, they create a plural word pattern with its corresponding singular word pattern, some of these patterns with examples are shown in Table 6.

Using hybrid approach, Hadni et al. (2013) proposed a stemming algorithm. They combine Khoja's stemmer (Khoja and Garside 1999), light stemmer and N-Gram based approaches. Through Khoja's stemmer, if a given word matches with the pattern, its relevant word is extracted as root. If given word does not match any pattern then using light stemmer prefixes and suffixes are removed respectively and derived word's length is verified. Finally, the stem-candidate partition into bi-gram and compare the list of the stem bi-grams with the bi-grams in the root file. The similarity between root bi-gram and stem-candidate is calculated by the distance measure and most similar root is output. If distance greater than 0 then return stem-valid, otherwise the original word is returned. They claim their stemmer approach better than Khoja Stemmer (Khoja and Garside 1999) and light stemmer. A morphological analysis technique to extract the root of Arabic word is presented by Abu-Errub et al. (2014). The proposed method first removes the prefixes and suffixes iteratively using prefix and suffix matrix after deletion the prefixes and suffix letters, if the derived word length is 3-character, then it will be stem word, if greater than 3 then it contains infixes. By using morphological weights, matrix infixes are deleted and extracted the root/stem as shown in Table 7.

Al-Omari and Abuata (2014) used mathematical rules and several relations between letters to define stemmer for Arabic Language. Using a word length, a mathematical formula is

constructed to define the starting point. For even word length, (word length)/2 and in case of odd (word length + 1)/2 is used to define the starting point. Then extracted a word that includes starting point letter with two neighbors (left and right neighbors) of starting point letter and check this word in the dictionary, if found, stop searching. Otherwise, shifting the right by one letter to get new words. This process continues until new word is found in the dictionary or no more letters exist to the right. If root is not found, then start shifting to the left by one letter and get the new word. This process continues either new word found in the dictionary or no more letters exists to the right.

Stemmer presented by Al-Kabi et al. (2015) operates in three phases that work in a sequence. In the first phase prefixes and/or suffixes are removed on the basis of prefixes and suffixes list (e.g. "مكتبات" to "مكتب"). In the second phase the output of the first phase is matched with the given verb pattern list. Root is extracted by deleting the matching letter between the derived word from phase one and the pattern, e.g. the output of the first phase is "مكتب" that matches with "مفعل" pattern and the stem is "كتب". In the third phase, the root of the given word is extracted, but in Arabic language, some letters that come at the end or the middle of the words, these letters are adjusted in this phase to get the correct root, e.g. the letter (Waaw with Hamza above, "ؤ") in the middle of the word should be transformed to (Alif, "ا"). El-Defrawy et al. (2015) proposed Context-Based Arabic Stemmer (CBAS). This stemmer also works in three phases: context matrix construction, roots generation, and root selection. In context matrix construction a window of n words is defined in which target word will be at *i*th position and rest of the surrounding words will be its context. Roots generation phase consists of three steps: words segmentation, pattern matching and dictionary validation. In word segmentation, entered words are split into prefix, infix and suffix using prefix and suffix lists. Then the infix obtained from prior step is matched with the pattern, if a match is found, the word is passed to the next step for validation. Dictionary validation step ensures that extracted stem is linguistically correct but linguistic correctness is not enough, more correct root is extracted in the next step. The last step is selection of root, for this the correlation between a word, and its context is measured by three different statistical methods: Pointwise Mutual Information (PMI), Positive PMI (PPMI), and Smoothed PMI (SPMI). The proposed algorithm (CBAS) uses a variation of PMI and Smoothed PMI (SPMI) to handle sparse matrices. SPMI achieved the highest accuracy of 81.5%, PMI and PPMI achieved accuracy was 78.84 and 79.49%, respectively.

Discussion

Till now, variety of Arabic stemmers have been reported yet no standard stemmer has emerged. Aljlayl and Frieder (2002) divided the Arabic stemmer into two classes: root stemming and light stemming. Light stemmer simply truncates the prefixes and/or suffixes, but does not deal with infixes (Larkey et al. 2002). In case of heavy stemming prefix and suffix are removed then match the pattern and derive the stem, e.g. Arabic word والكاتب after removing prefix وال, matching the pattern فاعل, كتب as a root is extracted (Aljlayl and Frieder 2002). Light stemmer gives better results in Arabic information retrieval as compared to root stemmer (Larkey et al. 2002; Aljlayl and Frieder 2002).Light stemmers are faster and more efficient, and hold specific features of the word, but some truncate the affixes that are actually part of stem such as the Arabic words لدين (two boys) after removing the affix the word لد (has no meaning) is extracted as root. Moreover, they do not deal with broken plural words and infixes. Root based stemmers may give better results in machine translation and in other fields of Natural Language Processing (NLP) however it is completely dependent on derived rules and dictionary in order to handle the newly discovered words.

Arabic stemmers use patterns and/or affix list to chop off affixes however, to validate the generated results some stemmers use dictionary and rest used patterns. Usually patterns (awzan) are used to identify the broken plural and truncate the infix letters to extract the singular form of the query words. Pattern matching has its own limitations. Researchers have used different evaluation measures for designed stemmers that may include manual, recall, precision and f-measure. Usually, in case of manual testing, used dataset is small, however a few of the researchers have used large size corpus to train and test stemmers. This is a major deficiency in performance evaluation and results standardization. Most of the stemmers report their results in comparison with Khoja's stemmer (Khoja and Garside 1999) however a few stemmer use more than one methods of evaluation. Table 8 provides the comparative listing of different Arabic stemmers.

2.3 Stemming in Persian language

Persian language is spoken in Iran, Afghanistan and Tajikistan mainly. The number of Persian speaking population is small as compared to Urdu speaking, however the work in NLP and related technologies is far ahead of Urdu mainly in Iran (Lakshmi and Kumar 2014). Various applications have also been developed that use stemming process as preprocessing phase (Saraee and Bagheri 2013; Sarabi et al. 2013; Mahmoodi and Varnamkhasti 2014).

Tashakori et al. (2002) presented first rule base stemmer for Persian language. It removes the prefix and suffix iteratively on longest matching approach until no letter left to remove. It also uses recording/partial matching techniques to return correct stem/root of the Persian word. Recoding is a context sensitive transformation of the form like AXC → AYC, where X is the input string and Y is transformed string. A and C specifies the context of the transformation.

Taghva et al. (2005a) proposed a rule base stemmer for Persian language. They defined the lists of suffixes and prefixes, and set the lower bound of a stem word at least three letters. Furthermore, they divided the list of suffixes into verb-suffixes, plural-noun-suffixes, possessive-noun-suffixes, other-non-suffixes, and other-suffix groups. For example, Persian word نرفتند (they did not go) has verb suffix ند and prefix ن, stemmer first identifies the affixes then truncate affixes and derived the stem رفت (went). Noun suffix is stacked in the following pattern.

$$\{Possessive\text{-}noun\text{-}suffix\}\{Plural\text{-}noun\text{-}suffix\}$$
$$\{Other\text{-}noun\text{-}suffix\} <Stem>$$

Deterministic Finite Automata (DFA)with 40 states is used to truncate the appropriate prefix and suffixes, and extract the stem word. For example, the Persian word خواننده هایمان (or singers) in which یمان is a possessive noun suffix, ها is a plural noun suffix and نده is other noun suffix, the stemmer removes the suffixes and extracted the stem word خوان (sing).

Another rule based stemmer is presented by Mokhtaripour and Jahanpour (2006) titled Farsi rule based stemmer. Prefix and suffix lists are created and suffixes list is divided into different sub categories: auxiliary suffix, possessive and auxiliary suffixes, possessive suffixes, plural suffixes, comparative suffixes and other suffixes. This stemmer consists of ten phases. In the first phase indefinite article/possessive suffixes are removed, then from phase two to seven auxiliary suffix, possessive and auxiliary suffixes, possessive suffixes, plural suffixes, comparative suffixes and other suffixes are removed respectively. In phase, eight signs of infinitive and in phase nine special end letters are removed. In final phase ten, prefixes are removed and stem is extracted.

Table 8 Comparison of Arabic stemming technique

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Khoja and Garside (1999)	Stemming Arabic text	Rule base	Prefix, suffix and stop words lists, pattern and root word dictionary	It produces a valid root as it is checked by words pattern	Depend Root dictionary
				Low error rate	Space required to store the prefix, suffix and patter and root words dictionary
				Include the stop words that improve the stem	Slow in execution time
				Suitable for linguistic purpose applications	
Chen and Gey (2002)	An Arabic stemmer	Machine translation (MT) based and affix removal	Set of prefixes and suffixes lists	Irregular plural words transform into singular correctly by clustering approach	It is depend on the corpus that is used in clustering
			Stop word lists	English to Arabic machine translated words reduce to stem	Light stemmer truncates the affixes without prior knowledge of linguist
				Include the stop words that improve the stem	Machine Translation Base and Light stemmers are two independent stemmers. Does not mention how they work together
				Better for information retrieve system because recall and precision is high and use in text mining applications	
				Cross languages performance is better	
				Remove prefixes and suffixes	

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Nwesri et al. (2005)	Stemming Arabic conjunctions and prepositions	Using grammatical rules	Set of prepositions and conjunctions, and Dictionary	Produces the best output because after the removal or replace the affixes stem word is checked in the dictionary	Depend on Lexicon
				Better for information retrieve and text mining application	Required a significant amount of space for creating dictionary
				Low error rate	Computation time is high because at every change word is check into the dictionary
Taghva et al. (2005b, a)	Arabic stemming without a root dictionary	Patterns base	Prefixes, suffixes and words pattern lists	Does not require dictionary	Required large space to store the prefixes suffixes and words pattern lists
				Effective as Khoja and Garside (1999), Ghwanmeh et al. (2009)	Pattern list is small
				Effective normalization	Slow execution time
				Better for text mining	It does not deal with more than 7 letters words
				High recall and precision rate, so good for information retrieve application	
Al-Kabi and Al-Mustafa (2006)	Arabic root based stemmer	Rule base	Prefix, suffix and pattern	It does not depend on dictionary to produce the stem	The space required to store the prefix, suffix lists and patters
				It reduces the vocabulary size	The small data set is used for evaluating the performance of stemmer

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
				Remove prefixes, suffixes and infixes where ever applicable	The affixes are associated with word patterns that sometimes stem wrongly
				Good for information retrieve applications	
Al-Shammari and Lin (2008)	The educated text stemmer (ETS) Arabic	Rule base	Stop words proceeding nouns, stop words proceeding verbs and both	Include the stop words that improve the stem	Time taking stemmer
				Reduce stemming errors	Grammatical knowledge base
				Decreasing the computational time and data storage	Very complex to implement
				Reduce the vocabulary size	
				Include the stop words that improve the stem	
				Claim better from Khoja stemmer	
				Dictionary free stemmer	
Goweder et al. (2008)	A hybrid method for stemming Arabic text	Hybrid	Suffixes and prefixes lists	Handle Arabicized word	Space required storing suffixes, prefixes and root words dictionary
			Root word dictionary	Good choice of linguistic purpose applications	Depend on the dictionary
				Low error rate	Time taking stemmer because every word is checked into the stem dictionary
				Also use in text mining applications	
				Remove appropriate prefixes, suffixes and infixes	

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
El-Beltagy and Rafea (2011)	An accuracy-enhanced light stemmer for Arabic text	Rule base	Prefix, Suffix Lists and Infixes removal rules list	Dictionary less Words pattern are also not used Handle the broken or irregular plural words Good for information retrieve (IR) and text mining applications Claimed better performance from a light stemmer (Larkey) Remove appropriate prefixes and suffixes too	Handling the broken or irregular plural words' rules are not complete as only up to word length 5, infixes rules are defined Slow execution as every word has to pass three stages
Ababneh et al. (2012)	Rule-based light stemmer for Arabic language	Rule Base	Prefix, suffix and pattern list	Avoid miss stemming as every word is checked with the pattern, then remove the affixes Reduce the size of the dictionary by stemming Broken plural are handled by the dictionary of patterns and its corresponding singular Good for information retrieve applications	Required large space to store prefix, suffix and pattern dictionary Time consuming every word check in word pattern list May suffer over stemming and under stemming Tested on small data sets

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
				Prefixes and suffixes are removed wherever applicable	
				Claim better results than Light Stemmer (Larkey) and Root-Extraction Stemmer (Khoja)	
Hadni et al. (2013)	Arabic stemmer based hybrid approach	Hybrid	Prefix and suffix	Instead of stem dictionary Bi-gram and the dice measure similarity are used to confirm the validity of the stem word	High computation time
			Words pattern list	Tested on Large data	It is very complex to combine three algorithms
				Better for text mining application	Space required for prefix and suffix lists and word patterns
				Infixes are handled by Khoja stemmer	
				Good for information retrieve applications	
				Remove appropriate prefixes and suffixes	
				Claimed better performance as compared to Khoja Stemmer, Light Stemmer, N-Gram	

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Abu-Errub et al. (2014)	Arabic roots extraction using morphological analysis	Rule base	Prefix and suffix lists	Remove prefixes, suffixes and infixes where ever applicable	The space required to store awazan
			Morphological weights matrix (Awzan)	It can also reduce the vocabulary size	May suffer slow execution because every word is searched in awazan lists
				Reliable stem because after stemming process resultant word compared with its appropriate morphological weight, and then the extra letters are deleted from the word and derived the root	Tested on small data sets
				Dictionary less stemmer	
Al-Omari and Abuata (2014)	Arabic light stemmer (ARS)	Rule base (linguistic and mathematics rules)	Prefix, suffix and stem word	Remove antefix, prefix, suffix, and postfix where ever applicable	The storage required storing prefix, suffix lists and dictionary
				Better for text mining application	Tested on small data sets
				Return valid stem	Depend on the dictionary, if a word is not present in the dictionary, it may stem incorrect or remain unstem
				Good for machine translation and linguistic purposes	Very complicated and time taking to execute

Table 8 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Al-Kabi et al. (2015)	A novel root based Arabic stemmer	Rule base and pattern base	Prefix, suffix and pattern	Infixes are removed efficiently	Space required storing rules and patterns
				Remove the prefixes and suffixes where ever applicable	Small data set
				Dictionary less stemmer	Difficult to find the entire pattern in the corpus
				Claimed better result from Khoja and Ghwanmeh stemmer	Linguistic knowledge is required
				Compress the size of the vocabulary	
				Fast in execution as consist only three steps	
El-Defrawy et al. (2015)	Cbas: Context based arabic stemmer	Hybrid	Prefix and suffix lists	Return true root of the query word	High storage space required
			Dictionary	Claimed better result as compare to Khoja, ISRI, and Tashphanye stemmers	High execution time Its performance depends on the window size that extracts target word from the context
				Better for information retrieve and machine translation	

Table 9 Some rules with example

Rules with Example	Stem word
اسم +ها پسر ها(boys)	پسر(boy)
جاندار+ان پسران(boys)	پسر(boy)

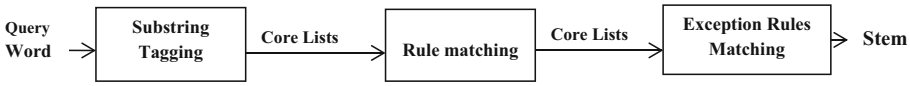

Fig. 2 Three phases of the proposed algorithm (Sharifloo and Shamsfard 2008)

Table 10 Suffix removal criteria

Length constrain	Recognized suffix for removal	Number of stemming rules base on the criteria
Word's length >6	Word's suffix equals "ترین" (in English "tareen")	2
	word's suffix equals "هایی" (in English "haayee")	4
Word's length >5	Word's suffix equals "های" (in English "haay")	2
	word's suffix equals "شان" (in English "shaan")	1

Sharifloo and Shamsfard (2008) published a hybrid stemmer in which rule based approach is combined with bottom up approach. They define 252 rules and 20 anti-rules. Rules are present in the form of cluster and every cluster is a set of morphemes. In the first step, entered word with length N is divided into $N \times (N + 1)/2$ substrings and the algorithm finds the substring that matches in morphemes list. After the first step, stem of given word will be found or morphemes which are derived from any stem, they call them cores. In step two, rules on cores are applied from the stored rules list to derive stem. If a rule is found, the word is returned as stem otherwise it is deleted from core list. Examples of rule base are given in Table 9. Step three, through exception list, stem extracted in the previous step is rectified e.g. the Persian word پرنده (bird) and پرنده ان is the wrong word, these are exceptional rules or anti rules. Diagrammatically working of this stemmer is shown in Fig. 2.

Rahimtoroghi et al. (2010) used a structural approach for Farsi stemmer. Based on Farsi structure of words and morphological rules, exceptions list is created. They describe the criteria for composing rules, i.e. word length and its corresponding suffixes. On the bases of these criteria stemming rules are developed. Rahimtoroghi et al. (2010) composed 36 rules for removing suffixes. For example, some criteria and its corresponding rules are mentioned in Table 10. Main goal of this stemmer is to improve the information retrieval performance so it deals with only noun, adjective and adverb and ignores the verb because verbs are rarely used in query.

Estahbanati et al. (2011) proposed another rule base stemmer for Persian language and claimed it is similar to the Porter stemmer. This stemmer used two lists: prefix list, suffix list where suffix list is divided as *verb-suffixes*, *plural-noun-suffixes*, *possessive-noun-suffixes*, *other-noun-suffixes* (e.g. نده), and *other-suffixes* (e.g. تر). For example, Persian word نرفتند (they

did not go) have verb suffix ند and prefix ن, the stemmer first identifies the prefix and suffix, then truncate affixes and derives the stem رفت (went). Noun suffixes have below given pattern (reading right-to-left):

$$\{Possessive\}\{Plural\}\{Other\} <Stem>$$

For example, the query word is ("our singers") "خوانندہ ہایمان", the stemmer first removes the possessive noun suffix "یمان" then removes the plural noun suffix "ہا" and after this finds and removes other-noun-suffix "ندہ", and extracts the stem خوان ("sing").

HPS (Hierarchical Persian Stemmer) presented by Rashidi and Lighvan (2014) works on two levels. In first level prefixes are chopped up by the prefix stemmer implemented by DFA (deterministic finite automata) machine. In second level suffixes are eliminated based on Part-Of-Speech (POS) tags (noun: N, adjective: A or verb: V). Each level has a hash table and a DFA (deterministic finite automata); hash table is used for exceptions and DFA (deterministic finite automata) to delete prefix and suffix. For example, the Persian word ننوشتیم (we did not write) contains prefix ن and suffix یم the stemmer chopped up the prefix and suffix and extract the stem نوشت (wrote). Noun suffixes have below pattern (reading right-to-left):

$$(Possessive)\,(Plural)\,(Other) <Stem>$$

For example, the query word is نوشتہ ہایمان (our writings), the stemmer first removes the possessive noun suffix "یمان" then removes the plural noun suffix "ہا" after this finds and removes the other-noun-suffix "ہ", and extracts the stem نوشت ("wrote").

Dianati et al. (2014) presented a language independent stemmer and tested on Persian and English. They use dictionary with clustered entries. This stemmer completes its execution in three steps: In step one, all the similar words with query word are selected from the clustered dictionary based on maximum matching in number and order of the letters. For example, two Persian (Farsi) words "درخت" (tree) and "درختها" (trees) are different in length but have the same stem. If query word is not similar to any word in relevant cluster, the new word (query word) itself is considered as a stem. In second step, the word chosen in step one is placed in the same cluster if it has the similar meaning; otherwise, it should be placed in different cluster. For example, the Persian words "کوچک" (small) and "کوچکتر" (smaller) are placed in the same cluster while the words "کوچ" (migration) and "چک" (check) are placed in different cluster. Similarly, the English words "teacher" and "teach" have the same cluster while "tea" and "each" are not in this cluster because their meaning is different. In the third step, after clustering the words, a stem corresponding to each cluster is determined. The stem derived by common and largest substring between words located in each cluster. For example, the Persian words "کوچک" (small) and "کوچکتر"(smaller), are in the same cluster and their stem is "کوچک" (small). In English language, a cluster contains two words "teacher" and "teach". Stem of this cluster will be the word "teach".

Rahimi (2015) used table lookup and affix removal technique to design stemmer. Rahimi uses Mokassar word list and intervening word list in table lookup approach and suffix list is used in the affix removal technique. First table looks up technique is used to get the stem of the given word. Initially look up in the Mokassar word list is made, if found return its corresponding singular form otherwise proceed and check the given word in intervening word list, if found, return the stem otherwise proceed the next step affix removal and extract the stem.

Taghi-Zadeh et al. (2015) proposed Persian stemmer that consists of two main parts: first, induction of stemming rules and second is applying these rules by a rule base stemmer to extract stem. Rule induction phase is further divided into three steps: in first step query

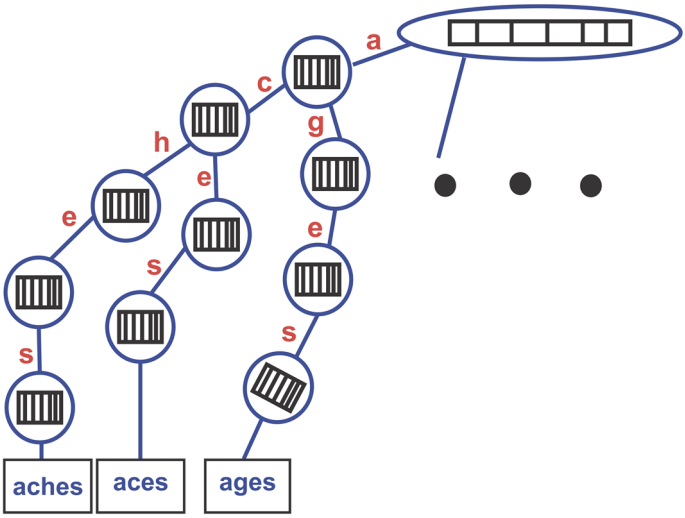

Fig. 3 Example of an inverted tree [produced from Taghi-Zadeh et al. (2015)]

words are converted into an inverted tree as shown in Fig. 3 that depicts an inverted tree data structure built for the three words 'aches', 'aces', 'ages' in 'e' and 'share two suffixes that are common in these words: 'aches', 'aces', 'ages'. Cut point used to extract the affixes from the inverted tree and is determined by two techniques: the maximum scores between the cut-points in each path of tree and second is maximum differences between the numbers of children nodes.

Maximum score: The cut-point in the path ($i_1, i_2, \ldots, i_k$), is the i-cutpoint which has the highest score.

$$i_{cutpoint} = \arg\max Score\,(i_x) \quad x \in [1..k]$$

Maximum score difference: Maximum score difference is calculated as:

$$i_{cutpoint} = \arg\max |Score\,(i_{x-1}) - Score\,(i_x)|\; x \in [1..k]$$

After this, extracted affixes are categorized into two sets: valid inflectional affix set and invalid inflectional affix set. The number of valid stems and the occurrence number of valid stems are two features that used to distinguish the valid and invalid affixes by clustering algorithm. Finally using extracted valid affixes, stemming rules are constructed in the form of regular expression.

Rule base stemmer is further divided into two steps: first is related to normalization i.e. standard query words and the second is to insure the validity of the stem by the dictionary.

Discussion

Persian stemmers usually use table look up, statistical and structural rule base techniques as discussed in Moghadam and Keyvanpour (2015). These three stemming techniques have their own limitations. To derive the stem of mokassar, Persian stemmers fully depend on stem dictionary. This dependency slows down the execution time and requires high storage space. Further, most stemmers are evaluated manually and none stemmer used (Paice 1994) and/or (Sirsat et al. 2013) stemmer evaluation methodology, some are tested on recall, precision and F-measure. Moreover, the size of testing data set is small and their claimed accuracy

may vary if the size of test datasets increased. No comparison has been made by any of the reviewed researcher to show the improvement in performance from prior presented stemmer.

In Table 11, we compare the different parameters of reviewed Persian stemmers.

3 Findings and conclusion

In this work, we review stemming approaches and identify the research gap in stemming approaches for Urdu and Urdu like languages. We also discuss the merits and demerits of each technique.

We found a number of issues with published Urdu stemmers. Firstly, all reviewed Urdu stemmers simply truncate the prefixes and/or suffixes and this kind of unguided removal of prefixes and suffixes causes over-stemming, under-stemming, or miss stemming especially where it is hard to distinguish between an affix letter and a root letter. Secondly, these stemmers do not deal with co-suffix like: صوبےدار سے صوبہ, thirdly, they do not change plural to singular like تدابیر سے تدبیر وغیرہ. Fourth issue is that these stemmers do not handle the composite words like کھانا وانا،بات چیت اور خاص خاص وغیرہ. Fifth issue is the handling of foreign words. For example, Persian words are used in Urdu as prefix: تہ (تہ بند) وغیرہ،پا(پامال) اور شاہ(شاہراہ،شاہرگ and Suffixes: الو (شرمالو،جھگڑالو) ،گار (مددگار) اور چہ،چی والے الفاظ وغیرہ

In Arabic language mostly stemmers are rule based and/or pattern based matching. These two techniques have their own limitations. For rule base stemmer, the linguistic knowledge of the language is compulsory; along with this almost every rule has its own exception cases. Moreover, some rules are difficult to implement in the form of computer understandable, e.g. broken plural words are got by pattern along with diacritics from the singular form. It is a challenging job to transform broken plural words to its corresponding singular form. Pattern matching has its own drawback and each pattern has its own exception list. Pattern matching is not sufficient itself instead it is used with other techniques. Sometime root stemmers give linguistically correct but wrong stem in reality as ولدين (two boys) by Khoja and Garside (1999) stemmer root لدن (soft) is extracted, it is linguistically correct but not valid stem, ولد (give birth) is valid stem (Nwesri et al. 2005). Moreover, some words do not belong to any pattern, handling such words is a difficult task. Patterns used in the Arabic root base stemmers are not listed completely. Although statistical and context sensitive approaches are also used to get better stem in Arabic language, but all the stemmers have their own benefits and drawbacks as a statistical approach gives advantage to not include the linguistic knowledge but it is completely dependent on the corpus that is used for training. Context sensitive stemmers' performance depends on the size of window to know the context and extract target words

Most of the Persian stemmers use structural rule base approach fully or partially, rules are developed on the bases of the morphology of Persian words. This type of stemmer simply removes the affixes until no affix is left to remove. This unguided removal causes under stemming and over stemming. This technique is efficient, easy to implement and requires less space. But in some cases linguistic rules become more complex. Further, prioritization of rules is also another issue in the structural rule base approach.

Although a lot of research work has been done to solve the problem of stemming, and to improve the accuracy of the stemmer in Persian and Arabic but a little work in Urdu is seen. Up to now used approaches for stemming in Urdu, Arabic and Persian are table look up, statistical approach and, most commonly used technique is a rule base approach. Each technique has its own merits and demerits. The table lookup approach is simple in use and

Table 11 Comparison of Persian stemming technique

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Tashakori et al. (2002)	Bon stemmer	Rule base	Prefixes and suffixes lists,	It truncate the affixes iteratively	Depend on dictionaries
			Dictionary of Persian infinitives, Mokassar Dictionary	Low error rate Better for linguistic purposes and text mining applications	Required a significant amount of space for creating dictionary
			Insensitive words dictionary and exception list	Plural words transform to singular	A lot of exceptions are used
Taghva et al. (2005b, a)	A stemming algorithm for the Farsi language	Rule base	Suffixes, prefix and exception list	It produce root of the word without dictionary	Tested on small corpus
				Good for information retrieve and text mining applications	It does not deal with prefixes
				Required less storage space as only suffix and exception lists stored	It does not handle broken plural words
				Suffixes are removed efficiently	High computation time, DFA represent the given word in Two-dimensional array
Mokhtaripour and Jahanpour (2006)	A new Farsi stemmer	Rule base	Prefixes and suffixes	It produce root of the word without dictionary	Infixes are not handled
				Less space required Suffixes and prefixes are removed according to given list of prefixes and suffixes Good for information retrieve (IR) and text mining applications	Time consuming every word will have to pass ten phases Small data set used to test the stemmer

Table 11 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Sharifloo and Shamsfard (2008)	Bottom up algorithm for Persian	Rule Base and Bottom up	Morpheme and rules list	Stem is reliable	Does not deal with infixes and irregular plural words
			Anti rules/exception list	Low error rate because after stemming every word is check in anti rules list	Time consuming every word has to pass three phases
				This stemmer can use in text mining applications	Space is required to store morphemes, rules and anti rules
				Appropriate prefix and suffixes are removed	
				Good for linguistic purposes application	
Rahimtoroghi et al. (2010)	A structural rule-based stemmer for Persian	Structural rule-based	Suffix list	Dictionary less stemmer	Space required storing exception lists
			Exception list	Fast in execution	Prefixes and infixes are not handled
				Good for text mining applications	Broken plural are not handle
				Low error rate	Difficult to define and implement the structure of the whole language
				Good choice for information retrieve application because high recall and precision rate	
				Remove suffixes where ever applicable	

Table 11 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Estahbanati et al. (2011)	A new method for stemming in persian Language	Rule Base	Prefix, suffix lists and Exception list	Dictionary less stemmer	Space required storing prefix, suffix and exception lists
				It also reduces the vocabulary size	Infixes are not handling
				Remove the prefixes and suffixes from the beginning and ending of the word	Slow execution because every word is check in exception list
				Good for information retrieve application (IR)	Small data set is used to test the stemmer
Rashidi and Lighvan (2014)	HPS: a hierarchical Persian stemming method	Hybrid	Prefix and suffix list	Dictionary less stemmer	Time consuming because every word first check in exception list
			Exception list	Compress the vocabulary	Infixes are not handled
				Remove appropriate prefixes and suffixes	The space required to stores prefix, suffix and exception lists
				This type of stemmer good for information retrieval	Broken plural words are ignored
					Small data from Hamshahri Collection is used to test stemmer
Dianati et al. (2014)	Words stemming based on structural and semantic similarity	Corpus base approach	Dictionary converted in clusters	Root base stemmer, words having a similar meaning placed in the same cluster	High execution time
				Tested on cross language	Large storage space required
				Better for Machine translation	Need to update data set, i.e. dictionary

Table 11 continued

Authors	Stemmers	Used approach	Required lists	Benefits	Limitations
Rahimi (2015)	A new hybrid stemming algorithm for Persian	Hybrid	Mokassar (broken plural) word list	It is a light stemmer	Its correctness is depending on the corpus
			Intervening word list	Broken plural words transform into its corresponding singular form	Requires weighty computing power for table lookup
			Suffix list	Better for text mining application	Prefixes are not handling
				Remove suffixes where ever applicable	Large storage space required
Taghi-Zadeh et al. (2015)	A new hybrid stemming method for Persian language	Hybrid	Stem dictionary	Auto rules extracted from inverted tree	Does not deal with infixes
				Remove prefix and suffix wherever applicable	High execution time
				No, exception word list used	Large storage space required
				Claimed better performance from Perstem, Dolamic and Taghva stemmers	
				Better for information retrieval and machine translation	
				No, exception rule list used	

gives accurate results. It is better choice if the language has no well-defined linguistic structure but problem with this approach is that if a word is not present in the database, it will remain unstemmed. Database needs to be updated to stem the newly discovered words, it requires large space, further it slows down the stemming process. A statistical approach may also be a good choice if a language has not well-defined linguistic rules or very complex linguistic rules that are impossible to translate in computer understandable format, linguistic expertise not required, it is language independent and can stem the word of new language with little effort. But problem with this approach is that its results are dependent on the data that is used during the training and testing. Its training set needs to be up-to-date, it also requires large space to store data set, moreover it slows down the execution of stemming. On the other hand, if a language has well defined grammatical structure, then rule based approach is better choice but none language has well defined rules and it is also a tiresome job to define and implement all the linguistic rules in computer understandable format because some grammatical rules are very complex.

To cover the above mentioned problems, hybrid approaches are being used but still, they are unable to assemble a standard stemmer. To date, there is no mechanism to handle the borrowed words from other languages and proper nouns in Urdu, Arabic and Persian. Moreover, researchers can study the stemming approaches in these three languages with the view that approaches coined for one language may support stemming process in the other two languages.

4 Recommendations

For Urdu language, there is a little work in stemming. Most of the stemmers are rule base stemmers with their own limitations. Context sensitivity is crucial to extract the true root of the query words, in some cases, linguistically extracted roots are invalid as compared to context sensitive roots. A stemmer can be assembled that consider the context of the stemable words and produce the stem accordingly. Porter approach has been implemented in many other languages; it can be tried on Urdu language. A cross language stemmer for Urdu language can be developed that may stem the other languages words after translation into Urdu. Word's feature extraction techniques (Seo et al. 2004) are better for morphological rich languages like Urdu. Self-learning techniques also need to be tested to develop Urdu stemmer that train itself from online or offline data, induce rules and using these rules stem the query word.

In Arabic language, various stemmers are constructed using different stemming techniques as mentioned above, but none stemmer is perfect stemmer that fulfills the information retrieval, machine translation and other requirements of Natural Language Processing (NLP) simultaneously. A lot of work has remained to be done in future to improve the stemmer performance. Self-learning techniques will be practiced to construct an Arabic stemmer that will use online data present in the form of web pages for training and testing. Context sensitive with words feature extraction techniques, eliminate the need of the window size that is used to extract the context and target word, give better stem and improve the execution time. Any combination of artificial intelligence techniques, e.g. classification, clustering techniques, machine learning, self-learning, pattern extraction and pattern matching techniques can be used to assemble the stemmer for any of three languages to fulfill most of the requirements of NLP applications.

Considering the quantity and quality of research, Arabic is ranked first and Urdu is ranked third. Persian language has certain rules for words creation (Estahbanati et al. 2011). For such languages words feature extraction (Seo et al. 2004) techniques will give better results. Combination of rules with context can also improve the Persian text stemming. Moreover, self-learning techniques can also be practiced on Persian stemmer. This stemmer may use off line created data set or online in the form of web pages.

Deep learning based approaches are found to produce promising results in computer vision, image processing, speech processing and natural language processing. Researchers may explore the use of these techniques for stemming purpose to achieve better results.

5 Our future work direction

In the future, we intended to develop an intelligent and language independent stemmer that can learn from data obtained through different sources i.e. Web Pages, language text repositories and articles. The adopted process will induce rules and using these rules stem extraction of query word will be made. For this, we intend to explore artificial intelligence techniques, machine learning techniques and pattern matching.

References

Ababneh M, Al-Shalabi R, Kanaan G, Al-Nobani A (2012) Building an effective rule-based light stemmer for Arabic language to improve search effectiveness. Int Arab J Inf Technol (IAJIT) 9(4):368–372

Abbas Q (2012) Building a hierarchical annotated corpus of urdu: the URDU. KON-TB treebank. In: International conference on intelligent text processing and computational linguistics. Berlin, pp 66–79

Abu-Errub A, Odeh A, Shambour Q, Hassan OA-H (2014) Arabic roots extraction using morphological analysis. Int J Comput Sci 11:2

Akram QA, Naseer A, Hussain S (2009) Assas-Band, an affix-exception-list based Urdu stemmer. In: Proceedings of the 7th workshop on Asian language resources. Association for Computational Linguistics, pp 40–46

Aljlayl M, Frieder O (2002) On Arabic search: improving the retrieval effectiveness via a light stemming approach. In: Proceedings of the eleventh international conference on Information and knowledge management. ACM, pp 340–347

Al-Kabi M, Al-Mustafa R (2006) Arabic root based stemmer. In: Proceedings of the international Arab conference on information technology

Al-Kabi M, Al-Shawakfa E, Alsmadi I (2013) The effect of stemming on Arabic text classification: an empirical study. Inf Retr Methods Multidiscip Appl 207–225

Al-Kabi MN, Kazakzeh SA, Ata BMA, Al-Rababah SA, Alsmadi IM (2015) A novel root based Arabic stemmer. J King Saud Univ Comput Inf Sci 27(2):94–103

Al-Omari A, Abuata B (2014) Arabic light stemmer (ARS). J Eng Sci Technol 9(6):702–717

Al-Shammari ET (2013) Lemmatizing, stemming, and query expansion method and system. U.S. Patent No. 8,473,279. 25 Jun 2013

Al-Shammari ET, Lin J (2008) Towards an error-free Arabic stemming. In: Proceedings of the 2nd ACM workshop on improving non English web searching. ACM, pp 9–16

Balakrishnan V, Lloyd-Yemoh E (2014) Stemming and lemmatization: a comparison of retrieval performances. Lect Notes Softw Eng 2(3):262–267

Cambria E, White B (2014) Jumping NLP curves: a review of natural language processing research. IEEE Comput Intell Mag 9(2):48–57

Carpineto C, Romano G (2012) A survey of automatic query expansion in information retrieval. ACM Comput Surv (CSUR) 44(1):1

Chen A, Gey FC (2002) Building an Arabic stemmer for information retrieval. In: TREC, pp 631–639

Chris DP (1990) Another stemmer. ACM. SIGIR Forum 24(3):56–61

Dahab MY, Al-Mutawa R (2015) A comparative study on Arabic stemmers. Change 125(8):

Dianati MH, Hadi SM, Rasekh AH, Fakhrahmad SM, Taghi-Zadeh H (2014) Words stemming based on structural and semantic similarity. Comput Eng Appl J 3(2):89–99

Ebrahim S, Hegazy D, Mostafa MG, El-Beltagy SR (2015) English–Arabic statistical machine translation: state of the art. In: International conference on intelligent text processing and computational linguistics. Springer International Publishing, pp 520–533

El-Beltagy Samhaa R, Rafea Ahmed (2011) An accuracy-enhanced light stemmer for arabic text. ACM Trans Speech Lang Process (TSLP) 7(2):2

El-Defrawy M, El-Sonbaty Y, Belal NA (2015) Cbas: context based arabic stemmer. Int J Nat Lang Comput (IJNLC) 4(3):1–12

El Kholy A et al (2013) Selective combination of pivot and direct statistical machine translation models. In: Proceedings of the 6th international joint conference on natural language processing

Estahbanati A, Javidan R, Dezfooli MA (2011) Implementation of a new method for stemming in Persian language. In: Proceedings of the international conference on web intelligence, mining and semantics. ACM, p 63

Frakes WB (1992) Information retrieval: data structures and algorithms, Chapter 8. http://orion.lcg.ufrj.br/Dr.Dobbs/books/book5/chap08.htm. Retrieved 1 Oct 2015

Ghwanmeh S, Kanaan G, Al-Shalabi R, Rabab'ah S (2009) Enhanced algorithm for extracting the root of Arabic words. In: Sixth international conference on computer graphics, imaging and visualization, 2009. CGIV'09. IEEE, pp 388–391

Goweder A, Alhami H, Rashed T, Al-Musrati A (2008) A hybrid method for stemming Arabic text. J Comput Sci. http://eref.uqu.edu.sa/files/eref2/folder6/f181.pdf

Gupta V, Joshi N, Mathur I (2013) Rule based stemmer in Urdu. In: 2013 4th international conference on computer and communication technology (ICCCT). IEEE, pp 129–132

Gupta V, Joshi N, Mathur I (2015) Design and development of rule based inflectional and derivational Urdu stemmer 'Usal'. In: 2015 international conference on futuristic trends on computational analysis and knowledge management (ABLAZE). IEEE, pp 7–12

Habash N (2007) Arabic morphological representations for machine translation. Arabic computational morphology. Springer, Netherlands, pp 263–285

Hadni M, Lachkar A, Alaoui OS (2012) A new and efficient stemming technique for Arabic Text Categorization. In: 2012 international conference on multimedia computing and systems (ICMCS). IEEE

Hadni M, Ouatik SA, Lachkar A (2013) Effective Arabic stemmer based hybrid approach for Arabic text categorization. Int J Data Min Knowl Manag Process (IJDKP) 3(4):1–14

Husain MS, Ahamad F, Khalid S (2013) A language independent approach to develop Urdu stemmer. Advances in computing and information technology. Springer, Berlin, pp 45–53

Hussain S (2008) Resources for Urdu language processing. In: IJCNLP, pp 99–100

Hussain S, Afzal M, (2001) Urdu computing standards: Urdu zabta takhti (uzt) 1.01. In: Multi topic conference, (2001) IEEE INMIC 2001, Technology for the 21st century. Proceedings, IEEE International, IEEE

Khan S, Anwar W, Bajwa U, Wang X (2015) Template based affix stemmer for a morphologically rich language. Int Arab J Inf Technol 12(2):146–154

Khan SA, Anwar W, Ijaz BU, Wang X (2012) A light weight stemmer for Urdu language: a scarce resourced language. In: 24th international conference on computational linguistics, p 69

Khansir AA, Mozafari N (2014) The impact of Persian language on Indian languages. Theory Pract Lang Stud 4(11):2360–2365

Khoja S, Garside R (1999) Stemming Arabic text 1999. http://zeus.cs.pacificu.edu/shereen/research.htm#stemming. Accessed 27 Dec 2015

Korenius T et al (2004) Stemming and lemmatization in the clustering of finnish text documents. In: Proceedings of the thirteenth ACM international conference on Information and knowledge management. ACM

Lakshmi RV, Kumar SBR (2014) Literature review: stemming algorithms for Indian and Non-Indian languages. Int J Adv Res Comput Sci Technol 2(3):349–352

Larkey LS, Ballesteros L, Connell ME (2002) Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis. In: Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval. ACM, pp 275–282

Lehal RKVGGS (2012) Rule based Urdu stemmer. In: 24th international conference on computational linguistics, p 267

Lovins JB (1968) Development of a stemming algorithm. Electronic Systems Laboratory, MIT Information Processing Group, Cambridge

Madnani N, Tetreault J, Chodorow M (2012) Re-examining machine translation metrics for paraphrase identification. In: Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: human language technologies. Association for Computational Linguistics

Mahmoodi M, Varnamkhasti MM (2014) Design a Persian automated plagiarism detector (AMZPPD). arXiv preprint arXiv:1403.1618

Majumder P, Mandar M, Swapan KP, Kole G, Mitra P, Datta K (2007) YASS: yet another suffix stripper. ACM Trans Inf Syst (TOIS) 25(4):18

Melucci M, Orio N (2003) A novel method for stemmer generation based on hidden Markov models. In: Proceedings of the twelfth international conference on information and knowledge management. ACM, pp 131–138

Moghadam FM, Keyvanpour M (2015) Comparative study of various Persian stemmers in the field of information retrieval. J Inf Process Syst 11(3):450–464

Mokhtaripour A, Jahanpour S (2006) Introduction to a new Farsi stemmer. In: Proceedings of the 15th ACM international conference on information and knowledge management. ACM, pp 826–827

Mubashir Ali SK, Saleemi MH (2014) A novel stemming approach for Urdu language. J Appl Environ Biol Sci 4(7S)436–443. ISSN: 2090–4274. www.textroad.com

Nwesri AFA, Tahaghoghi SMM, Scholer F (2005) Stemming Arabic conjunctions and prepositions. International symposium on string processing and information retrieval. Springer, Berlin, pp 206–217

Paice CD (1994) An evaluation method for stemming algorithms. Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval. Springer, New York, pp 42–50

Piotrowski M (2012) Natural language processing for historical texts. Synth Lect Hum Lang Technol 5(2):1–157

Porter MF (1980) An algorithm for suffix stripping. Program 14(3):130–137

Rahimi A (2015) A new hybrid stemming algorithm for Persian. arXiv preprint arXiv:1507.03077

Rahimtoroghi E, Faili H, Shakery A (2010) A structural rule-based stemmer for Persian. In: 2010 5th international symposium on telecommunications (IST). IEEE, pp 574–578

Rashidi A, Lighvan MZ (2014) HPS: a hierarchical Persian stemming method. arXiv preprint arXiv:1403.2837

Sarabi Z, Hamidreza M, Mojgan F (2013) Parsi Pardaz: Persian Language Processing Toolkit. In: 2013 3rd international conference on computer and knowledge engineering (ICCKE). IEEE

Saraee M, Bagheri A (2013) Feature selection methods in Persian sentiment analysis. International conference on application of natural language to information systems. Springer, Berlin, pp 303–308

Seo Y-W, Ankolekar A, Sycara K (2004) Feature selection for extracting semantically rich words. No. CMU-RI-TR-04–18. Robotics Inst., Carnegie-Mellon Univ., Pittsburgh

Sharifloo AA, Shamsfard M (2008) A bottom up approach to Persian stemming. In: IJCNLP, pp 583–588

Sirsat SR, Chavan V, Mahalle HS (2013) Strength and accuracy analysis of affix removal stemming algorithms. Int J Comput Sci Inf Technol 4(2):265–269

Taghi-Zadeh H, Hadi SM, Diyanati MH, Rasekh AH (2015) A new hybrid stemming method for Persian language. Digital Scholarship in the Humanities: fqv053

Taghva K, Beckley R, Sadeh M (2005a) A stemming algorithm for the farsi language. In: Null. IEEE, pp 158–162

Taghva K, Elkhoury R, Coombs J (2005b) Arabic stemming without a root dictionary. In: Innull. IEEE, pp 152–157

Tahir N (2014) Impact of Arabic language on Urdu language. VFAST Trans Islam Res 5(1):1–13

Tashakori M, Meybodi M, Oroumchian F (2002) Bon: the Persian stemmer. EurAsia-ICT 2002: information and communication technology. Springer, Berlin, pp 487–494

Zughoul M, Abu-Alshaar A (2005) English/Arabic/English machine translation: a historical perspective. Transl J 50(3):1022–1041