Journal of Experimental & Theoretical Artificial Intelligence

ISSN: 0952-813X (Print) 1362-3079 (Online) Journal homepage: https://www.tandfonline.com/loi/teta20

An improved Urdu stemming algorithm for text mining based on multi-step hybrid approach

Abdul Jabbar, Sajid Iqbal, Adnan Akhunzada & Qaisar Abbas

JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE
2018, VOL. 30, NO. 5, 703-723
https://doi.org/10.1080/0952813X.2018.1467495

Taylor & Francis
Taylor & Francis Group

An improved Urdu stemming algorithm for text mining based on multi-step hybrid approach

Abdul Jabbar[a], Sajid Iqbal[b], Adnan Akhunzada[a] and Qaisar Abbas[c]

[a]Department of Computer Science, COMSATS Institute of Information Technology (CIIT), Islamabad, Pakistan; [b]Department of Computer Science, Bahauddin Zakariya University, Multan, Pakistan; [c]College of Computer and Information Sciences, Al Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia

ABSTRACT

Stemming is the basic operation in Natural language processing (NLP) to remove derivational and inflectional affixes without performing a morphological analysis. This practice is essential to extract the root or stem. In NLP domains, the stemmer is used to improve the process of information retrieval (IR), text classifications (TC), text mining (TM) and related applications. In particular, Urdu stemmers utilize only uni-gram words from the input text by ignoring bigrams, trigrams, and n-gram words. To improve the process and efficiency of stemming, bigrams and trigram words must be included. Despite this fact, there are a few developed methods for Urdu stemmers in the past studies. Therefore, in this paper, we proposed an improved Urdu stemmer, using hybrid approach divided into multi-step operation, to deal with unigram, bigram, and trigram features as well. To evaluate the proposed Urdu stemming method, we have used two corpora; word corpus and text corpus. Moreover, two different evaluation metrics have been applied to measure the performance of the proposed algorithm. The proposed algorithm achieved an accuracy of 92.97% and compression rate of 55%. These experimental results indicate that the proposed system can be used to increase the effectiveness and efficiency of the Urdu stemmer for better information retrieval and text mining applications.

ARTICLE HISTORY
Received 2 June 2017
Accepted 3 April 2018

KEYWORDS
Urdu; affixes; lemmatization; stemming; natural language processing; Urdu stemmer; text mining; information retrieval

1. Introduction

Stemming is a process in which affixes are chopped up from derivational and inflectional forms to obtain the stem of the word. For example, Urdu words بااخلاق [Ba-akhlaq], اخلاقیات [akhlaqiyat] in which با [Ba] and یات [Yat] are affixes and their common stem is اخلاق [akhlaq]. Paice (1994) suggested that affixes should be removed to get the root word, and 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) define the stemming algorithm in Arabic language context as 'Stemming is the process of removing all of a word's prefixes, suffixes, and infixes to produce the stem or root'.

Conferring from recent studies, the NLP applications utilize the bag-of-words model that breaks the input text stream into uni-grams known as features. Each word is defined as a feature in the bag-of-words model. If there are multiple morphological forms of a word, then it contributes more features. However, if each inflectional or derivational form is reduced to its stem, the features are minimized

CONTACT Adnan Akhunzada a.qureshi@comsats.edu.pk

Table 1. Different shape of Urdu character.

Final	Medial	Initial	Letter
ع،ع	ـعـ	عـ	ع
نفع [Naffa]	تعارف	عابد	مثالیں
محل وقوع [Mahl-e-waqo]	[Tauraf]	[Abid]	[Misalain]

Table 2. English words with prefix and suffix.

Word	Prefix	Root	Suffix
unreadable	un	read	Able

and results are obtained with a little computation. The benefit of stemming is a multidimensional like reduction of features. These benefits and requirements are also required for the development of new type of algorithms like optimization of features through principle component analysis (PCA), the design of rule-based approaches (that are good for small features), case-based reasoning and other approaches besides machine learning approaches. These approaches are best suitable for a large number of features.

Stemming role differs from application to application; it improves the performance of information retrieval (IR) system as reported by many researchers such as (Flores & Moreira, 2016; Frakes & Baeza-Yates, 1992), machine translation (MT) system (Fattah, Ren, & Kuroiwa, 2006), and sentiment analysis (SA) (Oraby, El-Sonbaty, & El-Nasr, 2013). In text mining (TM), it reduces the document index size (Frakes & Baeza-Yates, 1992), in text summarization (TS) (Suman, Maddu, Shalini, & Bhavana, 2015) and text classification (Hadni, Ouatik, & Lachkar, 2013), stemming also reduces the number of features.

Literature review shows that three stemming approaches have been used for Urdu languages, i.e. rule-based approach (Akram, Naseer, & Hussain, 2009), statistical (N-Gram) approach (Husain, Ahamad, & Khalid, 2013) and template matching approach (Khan, Anwar, Bajwa, & Wang, 2015). Yet, all these stemmers suffer from basic deficiencies. To eliminate the deficiency of prior stemmers and to improve the accuracy of stemming process an efficient multi-step hybrid stemmer is proposed for the Urdu language.

The remainder of this paper is organized as follows. Section 2 describes the morphological aspect of Urdu language and Section 3 presents the literature review. Section 4 is dedicated to resources development and design of proposed stemming algorithm. Whereas Section 5 discusses the experimental results in terms of evaluation criteria, and time and space complexity of proposed stemmer. In the end, we presented conclusion and future works in Section 6.

2. Background

Urdu is highly Persianised/Arabicised (Islam, 2012; Parveen, 2008) and very different from other languages such as English. It is due to the fact that it has different rules of linguistics and phonetics. It was developed in the 12th century from the region Apabhramsha of northwestern India. Unlike most other Westerns languages, the writing orientation of Urdu script is from right to left. In Urdu, the word alphabets are connected and do not start with a capital letter as in the English language. Moreover, there are many different characters in shape based on their position in the word and adjunct letters. Table 1 demonstrates some of the Urdu character changes in shape.

In English, each word is separated by spaces but in Urdu, some words are not separated by space such as the word ادلےکابدلا [Adly ka badla]. Due to this reason, the segmentation step is a very challenging task in the Urdu language. In Urdu, the proper nouns do not start with a capital letter as in English, which makes it a challenging task for computerize machines to recognize and extract the proper nouns from Urdu texts. Furthermore, in English, the words are formed by attaching prefixes and suffixes to either or both sides of the root. For example, the word unreadable is formed as in Table 2.

Table 3. Example of infixes 3.

Urdu word	Infix	Root
موسر [Rasm]	و [vao]	مسر [Rasm]
ربکا [Akbar]	ا [alif]	رباکا [Akabar]

Table 4. Examples of broken plurals 4.

Singular	Plural
رفاک [kafr]	رافک [Kuffar]
روتسد [Dastoor]	ریتاسد [Dsatir]
نوتاخ [Khatoon]	نیتاوخ [Khawateen]

In English, affix letter/letters usually presented at the beginning or at the end of the word, but in the Urdu affix, a letter may be found in the middle of the word. This is called infix. This causes a serious issue in stemming Urdu documents because it is hard to differentiate between root characters and affix the letters. In Table 3, an example of infix is given.

In the Urdu language, a singular to plural transform is same as in English by adding suffixes and number of suffixes may be chosen from a list of suffixes like باتک[kitaab], ںیباتک[kitaben], ںوباتک[kitabon], and in case of broken plural words there is more variation, because they do not follow the normal morphological rules. Urdu broken plural words are somewhat like irregular English plurals. The difference is that English singular and plural words have a resemblance to each other, but in case of Urdu; both are non-identical as shown in Table 4.

3. Related works

Porter's stemmer is very famous due to its effectiveness in an information retrieval system (Porter, 1980) and most of the current stemmers are modified form of popular Porter's (1980) stemmer (Bimba, Idris, Khamis, & Noor, 2016). According to Kraaij and Pohlmann (1995) an effective stemming algorithm necessary for an efficient information retrieval for a complex morphological language. Researchers are developing and improving stemming algorithms in various languages such as English, Arabic, and Persian. We have reviewed all these stemmers and briefly described them in the following subsequent subsections.

3.1. English stemmers

Lovins (1968) proposed a context-sensitive, longest-match stemming algorithm with exception list for the English language. It consists of two steps, in first step affixes are truncated using the list of sixty rules, and in step second recoding procedure is performed to produce stem as a valid word. Porter (1980) presented a root base stemmer without exception list. It does not deal with prefixes. It iteratively removes the suffix until the termination condition meets. It has five steps: step 1 deal with plural and past participants. In step 2, 3, and 4 using specific rule suffix is stripped and in step 5 recoding is performed and return stem/root. Dawson (1974) proposed the non-iterative algorithm and extended the Lovins stemmer. It uses 1200 suffixes in a comprehensive way. Suffixes are arranged according to the longest length. It is very fast due to non-iterative but complex due to its comprehensive list of suffixes (Jivani, 2011). Majumder et al. (2007) presented YASS (Yet Another Suffix Striper) Stemmer. In this method, an equivalent, group/classes of words are created using a hierarchical clustering algorithm based on a measure of distance between strings. Each group is expected to represent an equivalence class consisting of morphological variants of a single root word. The words within a cluster are stemmed to the 'central' word in that cluster. The performance of this stemmer with respect to recall and precision is not good (Jivani, 2011).

3.2. Arabic stemmers

Taghva et al. (2005) presented an Arabic stemmer without a root dictionary. Various word patterns are defined and affixes (prefixes and suffixes) are removed using the longest match approach. The affixes are removed up to stem length of three characters. Al-Shammari et al. (2008) proposed the Educated Text Stemmer (ETS) for the Arabic language. This stemmer marks verbs and nouns in a query sentence using stop word list and applies the light stemming algorithm on nouns and Khoja's root based stemmer on verbs. Goweder et al. (2008) presented Arabic stemmer using a hybrid approach that consists of affix removal process, dictionaries, and morphological analysis techniques. In affix removal technique, affixes are removed according to the predefined list of prefixes and suffixes, and predefine rule pattern that is given below:

[Suffix Replacement].[Suffix].[Infix Replacement].[Infix].[Prefix eplacement].[Prefix].[WordLength]

Dictionary technique handles the improper removal of prefixes and morphological analysis techniques help to extract the root from broken plural words. Elrajubi (2013) proposed an Arabic stemmer that works in eight steps, in each step the specific length of affixes (prefix/suffix) are removed to derive the stem of the query word. Al-Kabi et al. (2015) have presented stemmer that operates in two phases. In the first phase, affixes (prefixes and suffixes) are truncated through a predefined list of affixes and in the second phase, the right root is extracted by pattern matching. Aldabbas et al. (2016) have developed prefix removal patterns and suffix removal patterns by a predefined list of prefixes and suffixes. Afterward, they constructed the regular expression on the basis of these patterns. The affixes are removed according to a regular expression. After this removal process, the meaning of both query word and stem are checked in the dictionary, if both have the same meaning, then the word is extracted as a stem. Abainia, Ouamour, and Sayoud (2016) proposed a stemming algorithm for the Arabic language, which consists of six steps. In the first step, some specific letters are replaced by others, for instance, various shapes of letter Alif أ, إ and آ are replaced by bare ا [Alif]. In step two, prefixes are truncated by predefined prefix list. The step three concerned to suffix removal process in which suffixes are eliminated with the help of a predefined list of suffixes. Whereas in step four, a plural Arabic word is transformed into singular form and step five transform a feminine noun to masculine. If the second, fourth, and fifth steps are failed then the query word may be a verb, and step six remove the prefix, suffix or both from the verb with a predefined list of verb prefix and suffix.

3.3. Persian stemmers

Tashakori, Meybodi, and Oroumchian (2002) proposed a first Persian rule-based stemmer called Bon. This stemmer iteratively removes the affixes until no more letters are left to delete as per defined criteria. After removing the affixes, the derived stem may be incorrect. To ensure the correctness, the stem recoding technique has applied. The recording is a context-sensitive transformation of the form AXC→AYC, where X is the input string and Y is transformed string, A and C specify the context of the transformation. Rahimtoroghi, Faili, and Shakery (2010) proposed a Persian stemmer, suffix identification, and deletion rules are developed by a morphological structure of the language. Rahimi (2015) presented the Persian stemmer using both table lookup and affix removal techniques. Table lookup approach is used to find the singular of Makassar words (broken plural words). When a query word is entered, firstly it is checked in the table, if found, then its corresponding stem is returned, otherwise, the affixes removal approach is used to derive the stem.

Table 5. Limitations of published Urdu stemmers.

Stemmer	Approaches	Limitations
Akram et al. (2009)	Rule base	• Unguided affixes are removed via a predefined list of affixes which sometimes leads to the erroneous stem • Infixes are ignored
Husain et al. (2013)	Statistical (N-Gram)	• A lot of unnecessary tokens are produced, leading to the exorbitant size of index files • Prefixes and infixes are ignored
Khan et al. (2015)	Template matching	• Almost all the patterns mentioned in this paper have some variations that do not handle, for example a pattern لوعف [Mafol] according to this pattern stem is derived by eliminating the third letter و [wao] as جورب [Barooj] stem to جرب [Barj] however the Urdu words روہظ [Zahoor], لوصح [Hasool], دوجس [Sajood], سولج [Jaloos] have the same pattern but cannot be stemmed in the same way • A provided pattern list is incomplete for instance لیعف، [Tafael] لیعفت [Fael] patterns are not included • Some Urdu words are patternless such as the تیم [Myat] and its root is توم [Moot]

3.4. Urdu stemmers

A little work is done in Urdu text stemming. Work found in literature can be categorized as rule-based, statistical (N-gram) and template matching methodologies. Akram et al. (2009) proposed the first stemmer with affix exception lists for Urdu text stemming, predefined affix lists are used to remove the affixes. Khan, Anwar, Bajwa, and Wang (2012)'s light stemmer is similar to (Akram et al., 2009) but does not consider the prefix and suffix rule exception lists. Lehal (2012) proposed Urdu stemmer that generates a list of possible stems from a predefined list of prefix and suffix. Urdu stemmer generates a list of possible stems if appropriate postfix/prefix rules are found, and then stem having high frequency is returned as root. Gupta, Joshi, and Mathur (2013) presented Urdu stemmer without exception lists, it also uses a list of predefined affixes, and if true affix is identified then the stem is derived after truncating the affix. Husain et al. (2013) presented a language independent stemmer and tested using Urdu language corpora, by using n-gram in which a set of consecutive letters that showed morphological variations are extracted as possible suffixes and these suffixes are deleted by using frequency based and suffix length-based approaches. It is found that length base approach gives better results as compared to frequency-based approach. Ali, Khlid, and Saleemi (2014) also proposed a rule-based stemmer for Urdu language, predefined affixes (prefixes and suffixes) list used to chop off affixes. Khan et al. (2015) stemmer use template matching approach to identify the infixes and if the template is matched, the appropriate infix letter is removed to extract stem. Gupta, Joshi, and Mathur (2015) improved their prior work (Gupta et al., 2013) and their improved stemmer checks the query words in exception words list and a stop word list, if not found then it applies the affix rules using a predefined affix list for stem extraction. A detailed survey of used stemming techniques in Arabic, Persian, and Urdu can be found in our survey paper (Jabbar, Iqbal, Ghani Khan, & Hussain, 2018).

3.5. Limitations of existing Urdu stemmers

According to our survey (Jabbar et al., 2018) about past studies, we found few papers on Urdu stemming in which some papers, use rule-based approach while only one paper utilizes the statistical method and other one uses the template based algorithm. Those approaches have their own deficiencies when they are used. A summary of those limitations is given in Table 5. In this table, we selected Akram et al. (2009)'s rule base stemmer as the representative of all the rules based stemmers because they claimed higher accuracy compared to other rule base stemmers.

To increase the performance of IR systems (Braschler & Ripplinger, 2004), they performed de-compounding and root extraction steps, but Urdu stemmers tokenized words are presented from input text by a hard space as a delimiter (Gupta et al., 2015; Lehal, 2012). In this way, the Urdu compound

Table 6. Developed language resources.

List name	Description
Prefix list (PL)	Contains 657 prefixes
Suffix list (SL)	Contains 614 suffixes
Reference lookup list (RLL)	Consists of 1364 entries
Stem words list (StWL)	Contains 40905 entries
Stop words list (SWL)	1148 words

words, i.e. ذمہ داری [Zima Dari] are split into two words, داری [Dari], ذمہ [Zima] and cannot be stemmed or wrong stemmed. We have also presented a method in this paper to extract the compound words from the Urdu text by using stop words technique.

It has been noticed from the review that the existing stemmers contain serious deficiencies as mention in Table 5. Moreover, their performance is below than acceptable level of many NLP applications. To overcome these limitations and enhance the performance of stemming process, we propose a multi-step hybrid approach that is based on three existing approaches: affix stripping rules (Akram et al., 2009) to eliminate the prefixes and suffixes while template matching (Khan et al., 2015) is used to trim infixes and by reference lookup (Bimba et al., 2016), variations of these rules are handled. The main role of references lookup approach is to avoid linguistic errors that may generate by the variation of rules as mentioned in Table 5. For example, Urdu word ظہور [Zahoor] produces three possible stems ظاہر [Zahir], ظہرہ [Zahirah], ظہر [Zuhar]. All of these produced stems are linguistically correct but the right stem is ظاہر [Zahir] that can be derived from the references lookup approach.

According to the past studies, the differences are seen in the structure of languages and most of the published stemmers are specific to a language (Ismailov, Abdul Jalil, Abdullah, & Abd Rahim, 2016). For instance, English stemmer simply uses an affix stripping and recoding rules to derive the stem. The mostly stemmer developer uses the Porter's (1980) English stemmer as a base to develop their stemmer (Bimba et al., 2016). Moreover, affix stripping stemmer is language dependent (Flores, & Moreira, 2016). For morphologically rich languages such as Urdu, Persian and Arabic, which utilized light stemmer, template matching, dictionary-based and unsupervised approaches. The morphological form of Urdu language is different as compared to English as described in Section 2. In this paper, we have developed an integrated approach for the Urdu stemming algorithm based on the affix stripping, template matching, and table lookup techniques.

4. Methodology

In this section, we present all the design and development steps of Urdu stemmer, which consists of two main phases such as resources development and design of stemming algorithm.

4.1. Resource development

To develop a good stemmer, it is essential to either build custom resources or use existing language resources to extract stem words from different derivational forms. These resources include affixes (prefixes and suffixes), reference lookup list/s (RLL), stop words (SW) list, stem word list (SWL). We surveyed and analyzed existing available resources and observed various limitations with existing resources like incomplete lists (Akram et al., 2009), the inclusion of unrelated words and data-set-specific lists (Burney, Sami, Mahmood, Abbas, & Rizwan, 2012). In this work, we have employed different language resources in the form of lists as given in Table 6. Detailed list of these resources available in (Jabbar, Iqbal, & Ghani Khan, 2016).

The affix lists ensure that all common affixes, inflectional affixes including noun plurals and derivational affixes, such as verbs and common nouns are incorporated affixes. All the selected prefixes and suffixes are arranged according to their length into 8 groups as shown in Tables 7 and 8.

Table 7. Example of Urdu prefixes

Length	Prefixes
One letter	ن، ب، ا، آ
Two letters	ان، کم، بن
Three letters	ابن، غیر، ہمہ
Four letters	ابدی، ابلق، اپی
Five letters	ابتدائی، ابالی، ابتدائے، میثاق
Six letters	گلوگیر، ابنائے، معروضی
Seven letters	آرائیاں، آہنگیوں، قرارداد، برادران
Eight letters	برادریوں، دستاویزی، مخالفانہ

Table 8. Example of Urdu suffixes.

Length	Suffixes
One letter	ے، ی، ا، ت، و
Two letters	اس، کی، تا، شک
Three letters	غیر، جگر، نما
Four letters	کاری، خانہ
Five letters	پزیری، روائی، ائیاں، ڈالہ، انگیز
Six letters	مزاحیہ، خداداد، افزائی، شاہانہ
Seven letters	وفاداری، **گزارانہ**، خودداری
Eight letters	سپرداری، جمالیاتی

Table 9. Urdu stop words.

Urdu	English
کا [Ka]	Of
پر [Par]	On
طرف [Traf]	To

One letter affix (prefix/suffix) is weak affix that must ensure before deletion, whether it is an actual part of a word or an affix. For instance, in Urdu word بخیریت [Bakheriat], ب [be] is a prefix and is deleted to extract the stem خیریت [Kheriat], but in case of بچپن [Bachman] the letter ب [be] is an actual part of the query word and could not be truncated.

For suffix such as اونچا [Oncha], ا [Alif] is truncated to get the stem اونچ [Onch] but in case of لڑکا [Larka] the letter ا [Alif] is an actual part of the query word and could not be truncated. After identification of such affix, if affix conations only one word, then this word is checked in the Stem Word List (SWL), if found, is returned, if not found, its prefix is removed to update the word that is rechecked in the Stem Word List (SWL), if found, is returned as stem otherwise returns original word.

Stop words are also common words that have a high frequency in the text, but are not used in information retrieval system (Atwan, Mohd, & Kanaan, 2013). Urdu stop word list contains postpositions, determiners, pronouns, and conjunctions (Gupta et al., 2015). On the other hand, content words are keywords of any sentence and have lexical meaning. Examples of stop words are shown in Table 9.

4.2. Corpus development for evaluation

We surveyed available corpora for testing our algorithm. Some developed corpora are not freely available (Hussain, 2008; McEnery, Baker, Gaizauskas, & Cunningham, 2000). Most of the developed corpora are application specific and lack the required variability (Ijaz & Hussain, 2007; Muaz, Ali, & Hussain, 2009). Moreover, all these corpora are content word based corpus.

For the above-mentioned reasons, we needed to develop our own corpus-based so we develop two corpus text fragments and word corpus. Text fragments, consisting of news articles (politics, literature,

Table 10. Example of suffix removal and recoding rules.

Criteria	Query words	Rule	No of Rules	Produce stem words	Right stem
The Word length >3 and ends with ے [ye]	وعدے [Wade]	Remove ے [ye]	3	وعد [Wad]	وعدہ [Wadah]
		Replace ے [ye] with ا [alif]		وعدا [Wada]	
		Replace ے [ye] with ہ [hay]		وعدہ [Wadah]	
	چرندے [Charinde]	Remove ے [ye]	3	چرند [Charind]	چرند [Charind]
		Replace ے [ye] with ا [alif]		چرندا [Charinda]	
		Replace ے [ye] with ہ [hay]		چرندہ [Charindah]	
	باجے [Baje]	Remove ے [ye]	3	باج [Baj]	باجا [Baja]
		Replace ے [ye] with ا [alif]		باجا [Baja]	
		Replace ے [ye] with ہ [hay]		باجہ [Bajah]	

science, and technology) collected from BBC Urdu [web1] and DAWN news [web2], containing 20000 words, including stop words, verbs, adverb, adjectives, nouns, proper noun, punctuation marks, English words, numbers and special symbols. Words corpus consists of 56074 words consisting of a uni-gram, bi-gram, tri-gram compound words, broken plural words and words with infixes are collected from four Urdu grammar books (Bloch, 2012; Haq, 1996; Board, 2010; UEP, 2014), resources provided by (Hussain, 2004) on Urdu morphology and online resources [Urdu online encyclopedia [web3], CLE Urdu words list [web4], CLE Urdu high frequency words list [web5].

4.3. Rule base development

Usually, the stem is obtained by removing the attached affixes (infixes and suffixes), but some affixes require substitution and/or addition along with deletion, for instance, وں، گی، ے and تا suffix requires deletion and substitution to extract the stem. So, we define the criteria on the basis of word's length and specific character at the specific location in the word. The example of handling such suffix rules is shown in Table 10.

The affixes are identified using predefined PL and SL if a true prefix/suffix or circumfix (both prefix and suffix) is found then appropriate affix is truncated to extract the stem. The affixes are chopped up in descending order, longest possible morpheme to the shortest morpheme. To avoid the under-stemming and over-stemming problems, we have removed affixes according to the length of query words. Minimum query word's length is set to four letters and minimum stem length is two characters, but if after removing the affix the derived stem's length is two letters, then this stem in looked up in the Stem Word List (SWL), if found, then returned as stem such as پل بندی [Pul Bandi]. The order of removing affixes is as follows: firstly remove circumfixes (both prefix and suffix) if found, and after this remove prefixes then suffixes are removed. For instance, in the Urdu word نا خوش گوار [Na Khush Gawar], نا [Na] and گوار [Gawar] are removed to extract the stem خوش [Khush]. The proposed stemmer ensures that during stemming, the gender of the word is not changed (example of such rules is given in Table 11) which makes it suitable to be used in different UNLP applications like machine translation (MT), parts of speech tagging (POS tagging) and automatic diacretization (Jabbar et al., 2016).

The patterns are developed on the information obtained from Urdu grammar books and published work (Bloch, 2012; Board, 2010; Hussain, 2004; UEP, 2014) to check the existence of infix, if found then its corresponding rules are applied to derive the stem.

The developed rule base consists of basic rules about words of specific length and their variations. For Urdu words of length four, we develop 10 rules along with five variations of each rule, 12 rules for Urdu words of length five along with eight variations of these rules and 13 rules for Urdu words of

Table 11. Example of suffix removal.

Criteria	Query words	Rule	No. of rules	Produce stem words		Stem word
If words length greater than 5 characters and end with ائیاں [aiyan]	خرچائیاں [Kharchian]	Remove اں [aan]	2	R1	خرچائی [Kharchai]	خرچ [Kharch]
		Remove ائیاں [ayyan]		R2	خرچ [Kharch]	
	مغلائیاں [Mughlaiyan]	Remove ائیاں [ayyan]	2	R1	مغل [Mughal]	مغلائی [Mugh-lai]
		Remove اں [an]		R2	مغلائی [Mughlai]	

Table 12. Infixes removal rules with variations.

Pattern	Query word	Rule	No of rules	Possible stem words	Right stem
The Word length is four, and the third letter is و [vao]	بروج [Barooj]	Remove third letter و [vao]	3	برج [Burj]	برج [Burj]
		Remove the third letter و [vao] and insert ا [alif] at position second		براج [Barj]	
		Remove third letter و [vao] and add ہ [hay] at the end.		برجہ [Barjah]	
	ظہور [Zahoor]	Remove third letter و [vao]	3	ظہر [Zuhar]	ظاہر [Zahir]
		Remove the third letter و [vao] and insert ا [alif] at position second		ظاہر [Zahir]	
		Remove third letter و [vao] and add ہ [hay] at the end.		ظہرہ [Zuhrah]	
	سجود [Sajood]	Remove third letter و [vao]	3	سجد [Sajd]	سجدہ [Sajdah]
		Remove the third letter و [vao] and insert ا [lif] at position second		ساجد [Sajid]	
		Remove third letter و [vao] and add ہ [hay] at the end.		سجدہ [Sajdah]	

length six along with one variation of these rules. Example of infix removal rules with variation is given in Table 12 and details are given in (Jabbar et al., 2016).

4.4. *Design of stemming algorithm*

The proposed approach for stemming Urdu text based on 1306 affix rules and predefined pattern for infixes. The algorithm works in 10 steps. Steps 1–7 are grouped as phase 1 and steps 8–10 are grouped as phase II. Phase-I deals with the extraction of unigram, bigram, and trigram words and could be considered as a preprocessing step. Affixes (prefixes, suffixes, circumfixes) attached with bi-gram and tri-gram is also removed in phase I. The result of the first phase is fed to phase-II for stem extraction if required where affixes (prefixes, suffixes, infixes, and circumfixes) attached with Uni-gram words are truncated by using affix lists (PL and SL). For instance, the Urdu sentence میں شادی شدہ ہوں [Main shadi shoda hun] is tokenized to extract the compound word شادی شدہ [shadi shoda] which is stemmed to شادی [shadi] in phase I. As another example, consider the Urdu sentence سیلابوں سے مویشیوں،انسانوں اور فصلوں کا 80٪ نقصان ہوا [silabo se maveshio,insano or faslun ka 80% nuqsan huwa], phase-1 tokens as سیلابوں [silabo], مویشیوں [maveshio], انسانوں

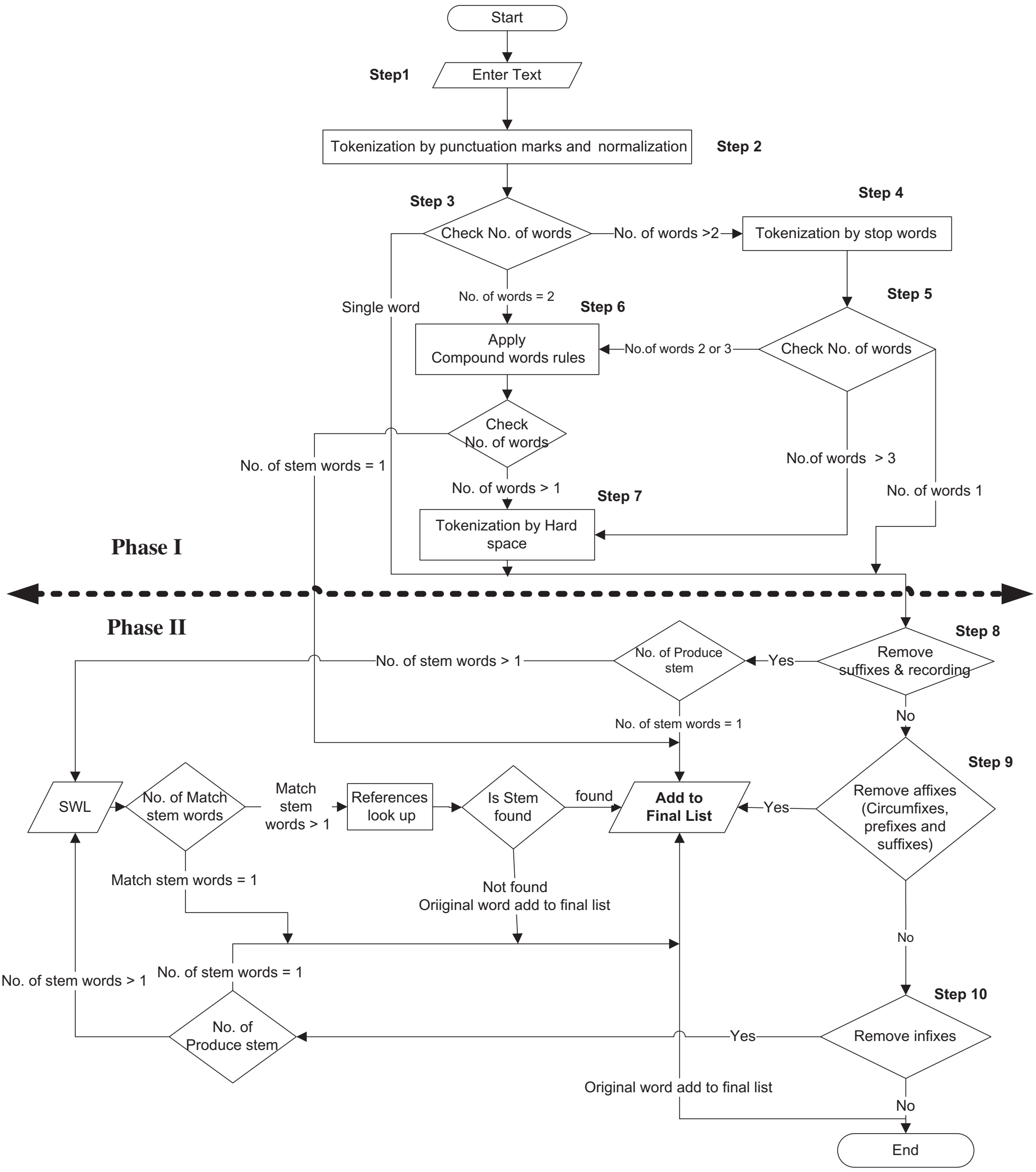

Figure 1. Flowchart of proposed Urdu stemmer.

[insano], فصلوں [faslun] then these token are fed into the phase II to apply appropriate rules to derive the stem.

In steps 6, 8 and 9, the affix stripping approach is applied, whereas the template-based approach is applied in step 10. Reference lookup approach is attached to the steps 8, 9 and 10. For instance, some suffixes and infixes may produce more than one possible stems, and in this case, possible stem is matched with SWL, if only one stem is matched, then it is returned, otherwise, references lookup approach is used and query word is searched in RLL, if found then its corresponding stem is returned. The algorithm and diagrammatic presentation of the stemming algorithm are shown in Figure 1 and explained below:

Algorithm:

Step 1: Read query text
Step 2: Tokenize by punctuation marks and normalize query text
Step 3: Check the No. of words in each token,
If the token contains greater than two words, then go to step 4
If the token contains two words, then go to step 6
If the length is one word, then go to step 8
Step 4: Tokenize by stop words
Step 5: Check the No. of words in each token,
If a token has a length greater than 3 words then go to step 7
If a token has 2 or 3 words, then go to step 6
If the token contains a single word, then go to step 8
Step 6: if compound words reduction rules found
Then extract the stem and add to the final list (FL)
Else go to step 7
Step 7: Tokenize the multi-gram by hard space
Step 8: If appropriate suffix removal and recoding rules found
Then extract the stem and add to the final list (FL)
Else go to step 9
Step 9: Check the true affixes by affix list (PL and SL) if found
Then truncate the affix and add the stem to the final list (FL)
Else go to step 10
Step 10: True infix is traced by predefined template if true infix found,
Then extracted the stem and added to the final list (FL)
Else add the original word to the final list

Algorithm description:

This system consists of ten steps, and each utilizes a specific resource and/or rules. We explain the working of the algorithm with the help of a running example. This algorithm executes in a linear fashion. Here, each of these steps is described in detail.

Step 1: Read the text or browse the file. An example is given in Table 13 (query text).

Step 2: To create a bag of words model, query text is tokenized. Tokenization is performed in three ways: first of all, the text is tokenized on the basis of punctuations (the used tokenization-marker list is given in Appendix 1), then on the stop word basis and finally, hard space is used as a delimiter. Tokenization by punctuation marks and normalization process is illustrated in Table 13. After tokenization and normalization, the algorithm proceeds to step 3.

Step 3: Depending upon the size of the token extracted in the prior step, the algorithm follows the specific route, if the token contains greater than two words, then step 4 is executed, if the length is two, then the next step is 6, if the token possess single word, then step 8 is operated.

Step 4: Tokenization by stop-words

As stated in the prior step, this step executed when the token obtained from step 2 contains more than two words, then there is the possibility of the presence of stop word because generally compound words are surrounded by stop words. The sentence is tokenized by the stop words, and proceeds to step 5, an example is shown in Table 14.

Step 5: After the step 4 there are three possible routes depending upon the size of token derived in the prior step. If a token contains two or three words, then step 6 is followed. If the token consists of

Table 13. Example of tokenization by punctuation marks and normalization.

Query text
تاریخ کے اوراق سیلاب کی تباہ کاریوں سے بھرے پڑے ہیں۔ سیلابوں سے مویشیوں، انسانوں اور فصلوں کا 80٪ نقصان ہوا ۔ ان حالات میں متاثرہ بزرگوں، نوجوانوں اور خواتین نے حکومت سے بروقت مدد کی اپیل کی ہے
Tokenized text
تاریخ کے اوراق سیلاب کی تباہ کاریوں سے بھرے پڑے ہیں
سیلابوں سے مویشیوں
انسانوں اور فصلوں کا
نقصان ہونا ہے
ان حالات میں متاثرہ بزرگوں
نوجوانوں اور خواتین نے حکومت سے بروقت مدد کی اپیل کی ہے

Table 14. Tokenization by stop-words

Prior tokenization by punctuation marks and normalization text
خیرات کے اوراق سیلاب کی تباہ کاریوں سے بھرے پڑے ہیں
Tokenized by stop words
خیرات
اوراق سیلاب
تباہ کاریوں
بھرے پڑے

a single word, then step 8 is followed, and if the token contains more than three words then algorithm moves to step 7.

Step 6: Compound word reduction

This step executed when the token size is two or three words either token obtained in step 3 or step 5. If input token contains two or three words, then possibly it is a compound word. For the token of size two words, it may contain suffix or prefix and if true affix is found, it is removed and the result is added to the final list (FL) as stem otherwise unchanged query word is passed to the next step. For example: **تباہ کاریوں [Tabah Karian] >> تباہ [tabah].** If the number of words in the token is three, then the existence of true circumfix (both prefix and suffix) is checked, for example غیر شادی شدہ [Ger Shadi Shuda], in which circumfix is traced and removed. The obtained result is added to the FL. If true circumfixes/s are not found, then step 7 is followed.

Step 7: Tokenization by hard space

Step 7 can be executed after either step 5 under the certain condition that is mentioned in step 5 or step 6. In this step the received token is split on the basis of hard space, and a list of single words is constructed.

Step 8: Suffix removal and recoding

Execution starts here if the specific condition meets in step 3 or step 5 or step 7. This process takes a word and checks the existence of the suffix on the basis of word length and specific character at the specific location in the word, if found, appropriate rules are applied. To check the application of rule variation, SWL is used. If produced word matches with more than one stem, then reference lookup table is used to find the most appropriate stem which is added to FL. If no suitable word is generated by this step, the original word is passed to next step 9. For example, it is clear from this word such as **لمحات [Lamhar] >> لمحہ [Lamha].**

Step 9: Affixes removal (circumfixes, prefixes, and suffixes)

If the stem of the query word is not derived in the prior step, then procedure initially checks the true circumfix (both prefix and suffix) by predefined PL and SL, true circumfix is removed if found to get the stem which is added to the FL. For instance: نوجوانوں [Nojawan] >> جوان [Jawan]. If true circumfix is not identified, then the true prefix is checked, if found, then the prefix is truncated and the result is added to the FL. For instance: بروقت [Barwaqat] >> وقت [waqat]. If a true prefix is not found, then check for the true suffix. If the true suffix is found, then remove the suffix and add a stem to FL, otherwise, the original word is passed to the next step 10. For example, these words are mentioned as بزرگوں [Bazurgun] >> بزرگ [Bazurg] and متاثرہ [Mutasrah] >> متاثر [Mutasr].

Step 10: Infixes removal

This procedure identifies the infix letters by using predefined patterns. If infixes are found, then its corresponding rules are applied, and the stem is extracted and added to FL. In case of no infixes are identified, then the original words are added to FL. The Urdu word خواتین [Khawateen] >> خاتون [Khatoon] is the best example.

5. Experimental results

As compared to affix removal stemmers, this algorithm can decrease the complexity of rule base because some rule variations can be handled by reference lookup table resulting in increased accuracy.

Table 15. Sensitivity and specificity of the evaluation method.

Stemmer evaluation metrics		
	Stem	Not stem
Words to be stem	True positive (TP)	False negative (FN
Words not to be stem	False positive (FP)	True negative (TN)

Stemming approaches based on dictionary lookup methods suffer from high storage requirements and not flexible for new words or variations (Hussain, Iqbal, Saba, Almazyad, & Rehman, 2017).

Our proposed algorithm uses a subset of the dictionary (stem word list) that eliminates the requirement of high storage and complex search techniques. As described in Section 4.2, two corpora are used such as word corpus and text corpus. For both corpora, a team of human experts is used to annotate the words along with right stems. Additionally, a team of experts has cross-validated the stems produced by each other. The designed stemmer is applied to raw data and stemmer results are compared with expert annotations to find the performance of the proposed algorithm.

To test both the morphological accuracy of stemmed Urdu words and effectiveness of our stemmer for information retrieval (IR) systems, a sequence of tests are carried out with two different types of corpora. Appendix 3 shows a sample taken from an actual stem and a stem produced by the proposed system.

Mainly three parameters are used to measure the performance of the proposed algorithm. These are stop word removal accuracy (SWRA), stemming accuracy (SA) and index compression factory (ICF).

Stop words removal accuracy: It can be calculated as:

$$\text{SWRA} = \frac{\text{No. of stop words removed}}{\text{Total stop words}} \times 100$$

Performance: Mainly the performance of a stemmer is measured using evaluation metrics given in Table 15 (Hadni et al., 2013).

Considering stated above parameters, we have evaluated proposed stemmer using following measures.

Accuracy: Accuracy indicates the ratio of the sum of truly stemmed words and truly un-stemmed words and a total number of words given to the system.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Recall: Recall indicates the ratio between a number of truly stemmed words by stemmer and the total possible stemmed words given to the stemmer. It is calculated as a percentage:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Precision: Precision is the ratio of the number of truly stemmed words by stemmer to the total words given to the stemmer. It is calculated as in percentage:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

F1-measure: The *F*-measure combines precision and recall. It is calculated as:

$$F_1(\text{recall}, \text{precision}) = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Confusion matrix statistics obtained after applying stemmer on both corpora is shown in Tables 16 and 17 respectively.

Finally, Table 18 shows the performance measures compared with our published works.

Table 16. Summary of words corpus results.

Words copus			
TP	50693	FN	457
FP	3486	TN	1438

Table 17. Summary of text corpus results.

Text corpus			
TP	10178	FN	268
FP	843	TN	198

Table 18. Accuracy, recall, precision and *F*-measure of test cases.

Stemmer	Corpus	No. of words	Used approach	Accuracy	Recall	Precision	*F*-measure
Our stemmer	Words corpus	56074	Hybrid	92.97%	99%	93.57%	96.26%
	Text corpus	20000		90.33%	97.43%	92.35%	94.82%
Akram et al. (2009)	Words corpus	21757	Rule base	91.2%	–	–	–
Husain et al. (2013)	Words corpus	1200	Statistics (N-gram)	84.27	–	–	–
Khan et al. (2015)	Words corpus	66200	Template matching	–	96.08	89.95	92.49

ICF: This represents the percentage of a collection of distinct words that is reduced by the stemmer, higher ICF shows greater the strength of a stemmer (Sirsat, Chavan, & Mahalle, 2013). Porter (1980) claimed his stemmer ICF to be 0.33, using 10,000 different English words. Our stemmer reduces the vocabulary size by 55%; this is because Urdu words have more morphological forms than English. According to Rizvi and Hussain (2005), an Urdu word may have sixty variants. ICF can be calculated as in percentage (Frakes & Fox, 2003) and is given as follows:

$$\mathrm{ICF} = \left(\frac{N - S}{S}\right) \times 100$$

where N = Number of distinct words before stemming; S = Number of distinct stems after stemming.

$$\mathrm{ICF} = \frac{56074 - 25233}{56074} \times 100$$

$$\mathrm{ICF} = 55\%$$

Figure 2 shows the ICF of proposed stemmer.

To evaluate our proposed approach, the stop word removal accuracy (SWRA), precision, recall, *F*-measure, and ICF were applied to word corpus contains 56074 words with a uni-gram (verbs, nouns, and adjectives), bi-gram, tri-gram compound words, broken plural words and words with infixes. We first checked the stop word removal accuracy (SWRA) by the following formula:

$$\mathrm{SWRA} = \frac{\text{No. of stop words removed}}{\text{Total stop words}} \times 100$$

The proposed stemmer achieved an admirable high accuracy of 98% along with this exception that if two words were not separated with hard space, such as انکےدوست [Un ke dost] then stemmer could not be able to distinguish between stop words and content words. In the case of words corpus, we obtained 92.97% accuracy, 99% recall, 93.57% precision and 96.26% *F*-measure, the rest of the accuracy can be achieved by proper identification of Urdu proper nouns and foreign words written in Urdu script, yet there is no mechanism in Urdu to identify them.

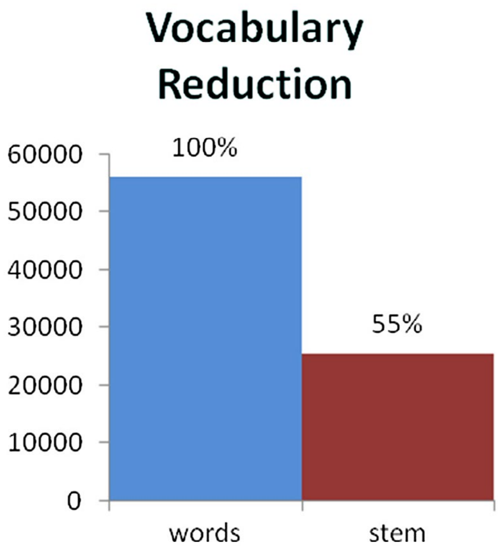

Figure 2. Index compression factor (ICF) of Urdu stemmer.

In case of text corpus, accuracy is low due to the improper segmentation and proper noun identification such as اور حکومت کے خلاف سخت نعرہ بازی کی [aur hukoomat ke khilaaf sakht naara baazi ki] after deleting the stop word حکومت خلاف سخت نعرہ بازی [hukoomat ke khilaaf sakht naara baazi] in this example نعرہ بازی [Narah Bazi] was compound word, but it is attached with other two words and has become four grams (خلاف سخت نعرہ بازی [ke khilaaf sakht naara baazi]) so it splits to a single word and considered as a single word that cannot be stemmed correctly.

The hybrid Urdu Stemmer is implemented as a sequential program with windows presentation foundation (WPF) user interface. We used C# .NET programming language for our multi-step hybrid stemming algorithm. This language is used because it has good string manipulation capability. The processing of the inflected word for stemming is done in three steps. The steps are mentioned below in details. In the first step, the inflected word is entered as an input. For example, the given paragraph is given as an input module.

تاریخ کے اوراق سیلاب کی تباہ کاریوں سے بھرے پڑے ہیں۔ سیلابوں سے مویشیوں، انسانوں اور فصلوں کا نقصان ہونا ہے۔ ماحول ناخوشگوار اور دردناک ہو جاتا ہے۔ لوگ بے گھر ہو جاتے ہیں۔ چوری چکاری بھی بڑھ جاتی ہیں۔ یہ لمحات زمینداروں کے لیے پریشان کن ہیں۔

[tareekh ke ourak selaab ki tabah kariyon se bharay parre hain. sailaabon se maweshion, insanon aur faslun ka nuqsaan hona hai. mahol nakhushgawaar aur dard naak ho jata hai. log be ghar ho jatay hain. chori chaakari bhi barh jati hain. yeh lamhaat zamindar ke liye pareshan kin hain](IPA translation and English translation of used Urdu words in this paper is given in Appendix 2)

This first input step is visually represented in Figure 3. In the second processing module, we applied prefix, suffix, infix, plural to singular transform and compound word reduction rules. After processing step, we developed an output module. This step is shown as in Figure 4.

The Time and space complexity are calculated based on different inputs. These efficiency measurements are displayed in Table 19 (Time complexity) and Table 20 (space complexity). From the Table 19, we got $O(n)$ time complexity if the user enters one word or compound words that must having affixes (prefix and suffix). We have also obtained $O(n^2)$ time complexity in the case of the text contains compound words and unigram words.

The space complexity of our proposed algorithm is constant time if the user enters one word. If user query a sentence, then its space complexity become linear, and required space for the algorithm increase proportional to the input size, as mention in Table 20.

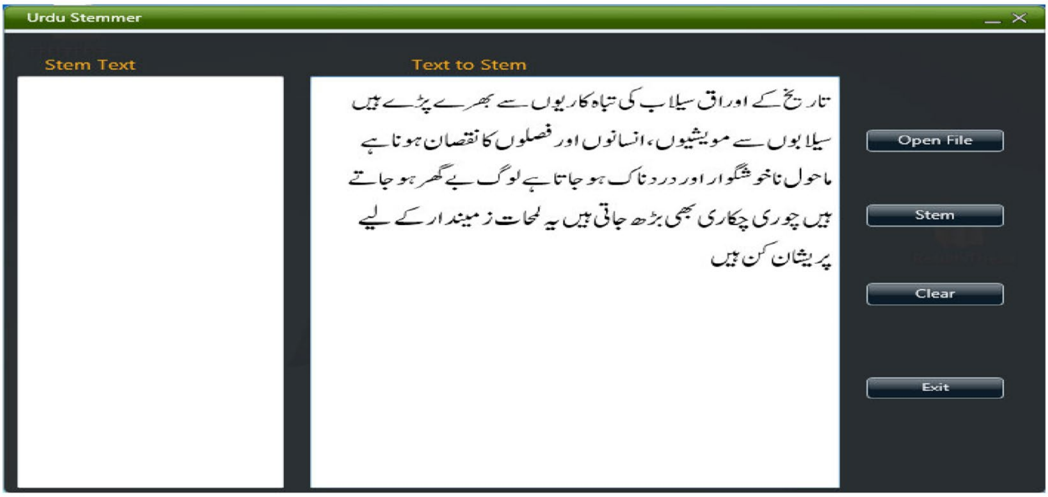

Figure 3. Graphical user interface design for input module.

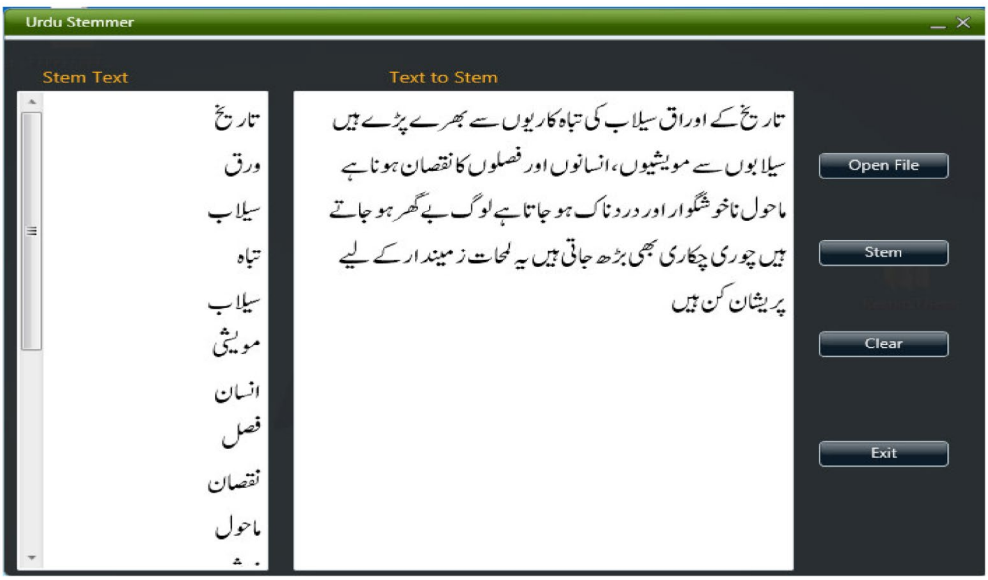

Figure 4. Interface design for output.

Table 19. Time complexity.

Cases	Time complexity	Reasons
Unigram or bigram words enter	$O(n)$	If we operate directly on unigram or bigram words and omit the functionality of extracting words from the text then its complexity reduces to $O(n)$
Text fragment enter may contain unigram, bigram words, stop words and non-Urdu words	$O(n^2)$	In this system, we handle the text that is why, the first loop is used to extract unigram or bigram words from the query text, and the second loop is used to derive the stem from the unigram or bigram words

Table 20. Space complexity.

Cases	Space complexity	Reasons
Unigram or bigram words enter	$O(1)$	In this case, one word is entered and its output also contains a word so space required for it is constant time
Text fragment enter may contain unigram, bigram words, stop words and non-Urdu words	$O(n)$	In this case, the input will be an array and output will also be in an array so its space complexity is linear type as increase the input data size, the space required for the algorithm is also increased

6. Conclusions

This system tokenize the text fragment into unigram, bigram and/or trigram words, and then extract the stem using a ten multi-step based hybrid approach that integrates three different stemming techniques i.e. affix stripping, template matching and table lookup in order to improve the effectiveness of the stemming process. Affix striping truncates the affixes (prefix and/or suffixes), template matching removes the infixes and reference lookup handles change stem errors. The performance of the proposed system was evaluated using two different valuation methods and two types of the corpora. We also compare the obtained results with the state-of-the-art Urdu stemmers (Akram et al. (2009); Husain et al. (2013); Khan et al. (2015)). The system achieved an accuracy of 92.97% and compression rate of 55%. The time complexity of the system is $O(n)$ in the case of word corpus, and $O(n^2)$ in the case of text corpus. The space complexity is constant time $O(1)$ on word corpus and on text corpus is linear type $O(n)$. Based on evaluation scores, we believe that proposed stemmer can be applied in many NLP applications like information retrieval (IR), text categorization (TC) and text mining (TM) with a good level of confidence.

Although results obtained from this research are better yet the following recommendations are identified for future work. To increase the accuracy of this stemmer, more context-sensitive and recoding rules may be added. One can also investigate more broken plural words by extracting the patterns for rules development. One can develop a method to identify proper nouns in the sentences, and improved segmentation rules in order to increase the accuracy of this stemmer. The limited size of the table looks up entries and Stem Word list (SWL) can be enhanced to increase the performance. Self-learning techniques like machine learning and latest emerging success of deep learning can also be tested to develop Urdu stemmer that trains itself from online or offline data, induces rules and by making use of these rules can stem the query word.

Acknowledgements

I would like to express my heartfelt thanks to the Masood Jhandir Research Library, Sardar pur Jhandir Mailsi, District Vehari, Punjab, Pakistan (jhandir.com) for providing us relevant, and important resources required for this research.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Abdul Jabbar http://orcid.org/0000-0001-8657-1282

References

Abainia, K., Ouamour, S., & Sayoud, H. (2016). A novel robust Arabic light stemmer. *Journal of Experimental & Theoretical Artificial Intelligence,* 1–17.

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

Al-Kabi, M. N., Saif, A. K., Belal, M. A. A., Saif, A. A.-R., & Izzat, M. A. (2015). A novel root based Arabic stemmer. *Journal of King Saud University-Computer and Information Sciences*, *27*(2), 94–103.

Al-Shammari, E. T., & Jessica, L. (2008, October). *Towards an error-free Arabic stemming. In Proceedings of the 2nd ACM workshop on Improving non english web searching* (pp. 9–16). ACM.

Aldabbas, O., Riyad, A.-S., Ghassan, K., & Mohammed, A. S. (2016). Arabic light stemmer based on regular expression. In *Proceedings of the International Computer Sciences and Informatics Conference (ICSIC 2016)*, 1–9

Ali, S., Khlid, S., & Saleemi, M. H. (2014). A novel stemming approach for Urdu language. *Journal of Applied Environmental and Biological Sciences, 4*(7S), 436–443.

Atwan, J., Mohd, M., and Kanaan, G. (2013). Enhanced Arabic information retrieval: Light stemming and stop words. In *Soft computing applications and intelligent systems* (pp. 219–228). Springer Berlin Heidelberg.

Bimba, A., Idris, N., Khamis, N., & Noor, N. F. M. (2016). Stemming Hausa text: Using affix-stripping rules and reference look-up. *Language Resources and Evaluation, 50*(3), 687–703.

Bloch, S. A. (2012). *“بنیادی اردو قواعد”,* مقتدرہ قومی زبان پاکستان, اسلام آباد. Islamabad: Muqtadra Qaumi Zabaan.

Board, P. T. (2010). *“اردو قواعد و انشاء” for Class-10th*. Lahore: Punjab Textbook Board.

Braschler, M., & Ripplinger, B. (2004). How effective is stemming and decompounding for German text retrieval? *Information Retrieval, 7*(3/4), 291–316.

Burney, A., Sami, B., Mahmood, N., Abbas, Z., & Rizwan, K. (2012). Urdu text summarizer using sentence weight algorithm for word processors. *International Journal of Computer Applications, 46* (19), 38–43.

Dawson, J. (1974). Suffix removal and word conflation. *Bulletin of the Association for Literary and Linguistic Computing, 2*(3), 33–46.

Elrajubi, O. M. (2013, November). An improved Arabic light stemmer. In *2013 International conference on research and innovation in information systems (ICRIIS)* (pp. 33–38). IEEE.

Fattah, M. A., Ren, F., & Kuroiwa, S. (2006). Stemming to improve translation lexicon creation form bitexts. *Information Processing & Management, 42*(4), 1003–1016.

Flores, F. N., & Moreira, V. P. (2016). Assessing the impact of stemming accuracy on information retrieval – A multilingual perspective. *Information Processing & Management, 52*(5), 840–854.

Frakes, W. B., & Baeza-Yates, R. (1992). *Information retrieval: Data structures and algorithms*. (Chapter 8: Stemming algorithms). Englewood Cliffs, NJ: Prentice Hall.

Frakes, W. B., & Fox, C. J. (2003). Strength and similarity of affix removal stemming algorithms. In *ACM SIGIR Forum* (Vol. 37, no. 1, pp. 26–30). ACM.

Goweder, A., Alhami, H., Tarik, R., & Al-Musrati, A. (2008). A hybrid method for stemming Arabic text. *Journal of computer Science*, URL: http://eref.uqu.edu.sa/files/eref2/folder6/f181.pdf.

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

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

Hadni, M., Ouatik, S. A., & Lachkar, A. (2013). Effective arabic stemmer based hybrid approach for arabic text categorization. *International Journal of Data Mining & Knowledge Management Process (IJDKP), 3*(4), 1–14.

Haq, M. A. (1996). *“قواعد اردو”,* انجمن ترقی اردو, *(ہند)* نئی دلی. New Dehli: Anjuman Taraqqi-e-Urdu.

Husain, M. S., Ahamad, F., & Khalid, S. (2013). A language Independent Approach to develop Urdu stemmer. In *Advances in computing and information technology* (pp. 45–53). Springer Berlin Heidelberg.

Hussain, Z., Iqbal, S., Saba, T., Almazyad, A. S., & Rehman, A. (2017). Design and development of dictionary-based stemmer for the Urdu language. *Journal of Theoretical & Applied Information Technology*, 95(15), 3560–3569.

Hussain, S. (2004). *Finite-state morphological analyzer for Urdu* (PhD diss.). National University of Computer & Emerging Sciences.

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

Ijaz, M., & Hussain, S. (2007, August). Corpus based Urdu lexicon development. In *The proceedings of conference on language technology (CLT07)*, University of Peshawar, Pakistan (Vol. 73).

Islam, R. A. (2012). *The morphology of loanwords in Urdu: The Persian, Arabic and English strands*. (Doctoral dissertation, Newcastle University). UK, Available online at URL: https://theses.ncl.ac.uk/dspace/bitstream/10443/1407/1/Islam%2C%20R.A.%2012.pdf [accessed 24/03/2016].

Ismailov, A., Abdul Jalil, M. M., Abdullah, Z., & Abd Rahim, N. H. (2016, August). A comparative study of stemming algorithms for use with the Uzbek language. In *2016 3rd International conference on computer and information sciences (ICCOINS)* (pp. 7–12). IEEE.

Jabbar, A., Iqbal, S., Ghani Khan, M. U., & Hussain, S. (2018). A survey on Urdu and Urdu like language stemmers and stemming techniques. *Artificial Intelligence Review, 49*(3), 339–373.

Jabbar, A., Iqbal, S., & Ghani Khan, M. U. (2016, November 18). Analysis and development of resources for urdu text stemming. In *Proceedings of the 6th annual international conference on language and technology, KICS-CLE*, UET Lahore.

Jivani, A. G. (2011). A comparative study of stemming Algorithms. *International Journal of Computer Technology and Applications, 2*(6), 1930–1938.

Khan, S. A., Anwar, W., Bajwa, U. I., & Wang, X. (2012, December). A light weight stemmer for Urdu language: A scarce resourced language. In *24th International conference on computational linguistics* (p. 69).

Khan, S., Anwar, W., Bajwa, U., & Wang, X. (2015). Template based affix stemmer for a morphologically rich language. *The International Arab Journal of Information Technology, 12*(2), 146–154.

Khoja, & Garside. (1999). *Stemming Arabic Text*. Lancaster, UK, Computing Department, Lancaster University. Available online at URL: http://zeus.cs.pacificu.edu/shereen/research.htm#stemming [accessed 27/12/2015].

Kraaij, W., & Pohlmann, R. (1995). Evaluation of a Dutch stemming algorithm. *The New Review of Document and Text Management, 1*, 25–43.

Lehal, R. K. V. G. G. (2012, December). Rule based Urdu stemmer. In *24th International conference on computational linguistics* (p. 267).

Lovins, J. B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics, 11*(1–2), 22–31.

Majumder, P., Mitra, M., Parui, S. K., Kole, G., Mitra, P., & Datta, K. (2007). YASS: Yet another suffix stripper. *ACM Transactions on Information Systems (TOIS), 25*(4), 18.

McEnery, A. M., Baker, J. P., Gaizauskas, R., & Cunningham, H. (2000). EMILLE: Towards a corpus of South Asian languages. *British Computing Society Machine Translation Specialist Group, 11*, 1–9.

Muaz, A., Ali, A., & Hussain, S. (2009, August). Analysis and development of Urdu POS tagged corpus. In *Proceedings of the 7th workshop on Asian language resources* (pp. 24–29). Association for Computational Linguistics.

Oraby, S., El-Sonbaty, Y., & El-Nasr, M. A. (2013). Exploring the effects of word roots for Arabic sentiment analysis. In *IJCNLP* (pp. 471–479).

Paice, C. D. (1994, August). An evaluation method for stemming algorithms. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 42–50). Springer-Verlag New York, Inc.

Parveen, A. (2008). *Morphological analysis of modern standard Urdu*. (Doctoral dissertation, Aligarh Muslim University). Aligarh, India.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*, 130–137.

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

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

Rizvi, S. M. J., & Hussain, M. (2005). "Analysis, design and implementation of Urdu morphological analyzer." In *Student conference on engineering sciences and technology, 2005. SCONEST 2005* (pp. 1–7). IEEE.

Sirsat, S. R., Chavan, V., & Mahalle, H. S. (2013). Strength and accuracy analysis of affix removal stemming algorithms. *International Journal of Computer Science and Information Technologies, 4*(2), 265–269.

Suman, M., Maddu, T., Shalini, A., & Bhavana, K. (2015). A new approach for text summarizer. *Compusoft, 4*(4), 1665.

Taghva, K., Elkhoury, R., & Coombs, J. (2005, April). Arabic stemming without a root dictionary. In Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on (Vol. 1, pp. 152–157). IEEE.

Tashakori, M., Meybodi, M., & Oroumchian, F. (2002). Bon: The persian stemmer. In *EurAsia-ICT 2002: Information and communication technology* (pp. 487–494). Springer Berlin Heidelberg.

UEP. (2014). *"تخلیق اردو گرائمر", for class 8th*. Urdu bazar Lahore: Unique Education Publisher.

W1. Retrieved from http://www.bbc.com/urdu

W2. Retrieved from https://www.dawnnews.tv

W3. Retrieved from http://www.urduencyclopedia.org/urdudictionary

W4. Retrieved from http://www.cle.org.pk/software/ling_resources/wordlist.htm

W5. Retrieved from http://cle.org.pk/software/ling_resources/UrduHighFreqWords.htm

Appendix 1

Token boundary marker list

{	}	(	)	[	]	>	<	-	_
=	+	|	\\	:	;	.	/	?	~
!	,	¡	"	’	’	÷	×	°	\
¿	@	#	$	%	^	&	*	ø	‘
؛	’	-	؟	’	0	1	2	3	4
5	6	7	8	9	A	B	C	D	E
F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y
Z	a	b	c	d	e	f	g	h	i
j	k	l	m	n	o	p	q	r	s
t	u	v	w	x	y	z	۰	۱	۲
۳	۴	۵	۶	۷	۸	۹			

Carriage return, Line feed and Form feed

Appendix 2

Urdu word	IPA translation	English translation
بااخلاق	/aːəxləq/	(Well mannered)
اخلاقیات	/ɑxlaqiat/	(Ethics)
اخلاق	/ɑxlaq/	(Well-behaved)
محلوقوع	/mhlʊɔqɔʔ/	(Location)
نفع	/nfa/	(Profit)
تعارف	/t̪ʔarf/	(Introduction)
عابد	/ʔabd̪/	(Devotionalist)
مثالیں	/msalĩ/	(Example)
رسوم	/rsum/	(Customs)
اکبر	/aɪkbr/	(great)
اکابر	/aɪkabr/	(the great)
رسم	/rsm/	Custom
کافر	/kafr/	(The disbeliever)
کفار	/kafar/	(The disbelievers)
دستور	/d̪st̪vr/	(Institution)
دساتیر	/d̪sat̪ivr/	(Institutions)
خاتون	/xat̪vn/	(Lady)
خواتین	/xvat̪jn/	(ladies)
بروج	/brvʤ/	(towers)
ظہور	/ʤheor /	(Manifestation)
حصول	/hsol/	(Acquisition)
سجود	/sʤʒod̪/	(Prostrate)
جلوس	/ʤlos/	(Promenade)
میت	/mit̪/	(Deceased)
موت	/mvt̪ /	(Death)
ذمہ داری	/zmh d̪arj/	(responsibility)
ظاہر	/zahr/	(Apparent)
ظہرہ	/zhra/	(Female Name)
ظہر	/zhr/	(The noon prayer)
بخیریت	/bxirit̪/	(with peace)
بچپن	/bʧpn/	(childhod)
اونچا	/ɑɛonʧa/	(high)
لڑکا	/lrka/	(Boy)
وعدے	/ɔʔd̪e/	(Promises)
وعدہ	/ɔʔd̪h/	(Promise)
چرندے	/ʧrnd̪e/	(The birds)
چرند	/ʧrnd̪h/	(The bird)
باجے	/baʤe/	(Musical instrument plural)
باجا	/baʤa/	(Musical instrument singular)
خرچائیاں	/xrʧaəiɑ̃/	(Costs)
خرچ	/xrʧ/	(Cost)
مغلائیاں	/mɣaəiɑ̃/	(Name of Caste)
مغلائی	/mɣaəi/	(Name of Caste)
غیر شادی شدہ	/ɣi ʃad̪I ʃɛd̪h/	(unmarried)
تباہ کاریوں	/t̪bah kariõ/	(Destructions)
تباہ	/t̪bah /	(Destroyed).
لمحات	/lmhat̪/	(moments)
لمحہ	/lmh/	(The moment)
نوجوانوں	/noʤʒoanõ/	(Teens)
جوان	/ʤʒoan/	(Young)
بروقت	/broqt̪/	(timely)
وقت	/oqt̪/	(time)
بزرگوں	/bzrgõ/	(elders)
بزرگ	/bzrg/	(elderly)
متاثرہ	/mt̪asrah/	(Affected)
متاثر	/mt̪asr/	(being affected).
ناخوشگوار	/na xoʃ goar/	(Unhappy)
خوش	/xoʃ/	(happy)
پل بندی	/pɛl bnd̪i/	(to construct a bridge)
میں شادی شدہ ہوں	/ miː̃ ʃad̪I ʃɛd̪h ɦʊ'ŋ/	(I am married)
شادی شدہ	/ ʃad̪I ʃɛd̪h/	(married)

(*continued*)

Appendix 2 (*Continued*)

Urdu word	IPA translation	English translation
سیلابوں سے مویشیوں، انسانوں اور فصلوں کا 80٪ نقصان ہوتا ہے	/ʂjbũ sæ mvjʃjũ, ɑnsɑnũ aor fslɛũ ka 80% nɛqsan hut̪a hæ/	(Floods cause the 80% loss of cattle, humans and crops)
سیلابوں	/ʂjbũ/	(floods)
مویشیوں	/mvjʃjũ/	(cattle)
انسانوں	/ɑnsɑnũ/	(humans)
فصلوں	/fslɛũ/	(crops)
نقصان	/nɛqsan/	(loss)

Appendix 3

Unstemmed word	Actual stem	System produce stem	Unstemmed Word	Actual stem	System produce stem
استری	استری	استری	پابند	بند	بند
پری میڈیکل	میڈیکل	میڈیکل	تدابیر	تدبیر	تدبیر
تھانے دار	تھانہ	تھانے	نسبتیں	نسبت	نسبت
غیرضروری	ضرور	ضرور	لازوال	زوال	زوال
زود فہم	فہم	فہم	فوائد	فائدہ	فائدہ
شکایات	شکایت	شکایت	عبارت آرائی	عبارت	عبارت
زود فہم	فہم	فہم	چوری چکاری	چور	چوری
خواتین	خاتون	خاتون	کتابیں	کتاب	کتاب
دلہن	دلہن	دلہن	تصورات	تصور	تصور
تکونیات	تکون	تکون	ڈبے	ڈبہ	ڈبہ
محدود	حد	حد	ہمہ وقت	وقت	وقت
آرائیاں	آرائی	آرائی	خوشگوار	خوش	خوش
تہ خانہ	تہ	تہ	سکول	سکول	سکول
نقاط	نقطہ	نقطہ	نوجوان	جوان	جوان
ان پڑھ	پڑھ	پڑھ	بیٹھک	بیٹھ	بیٹھ
امراض	مرض	مرض	دائروں	دائرہ	دائرہ
نمبرز	نمبر	نمبر	درپردہ	پردہ	پردہ
موجودہ	موجود	موجود	مساجد	مسجد	مسجد
مساواتیں	مساوات	مساوات	بااخلاق	اخلاق	اخلاق
سائنسدانوں	سائنس	سائنس	ہتھیاروں	ہتھیار	ہتھیار
زراعتیں	زراعت	زراعت	غضبناک	غضب	غضب
بےاعتباری	اعتبار	اعتبار	مردانگی	مردانہ	مردانہ
ضرورۃ	ضرور	ضرور	عاجزیوں	عاجز	عاجز
اساتذہ	استاد	استاد	نوعمر	عمر	عمر
ترتیب	ترتیب	ترتیب	جسمانی	جسم	جسم
پیدائش	پیدا	پیدا	حکیم نی	حکیم	حکیم
پائیہ تکمیل	تکمیل	تکمیل	بچاؤ	بچ	بچ
اراکین	رکن	رکن	جیل خانہ جات	جیل	جیل خانہ
زیربار	بار	بار	طالبات	طلبہ	طلبہ
مائعات	مائع	مائع	خرچائیاں	خرچ	خرچ
قوانین	قانون	قانون	مہاراج	راج	راج
مغالئیاں	مغالئی	مغالئی	اتحاف	تحفہ	تحفہ
ہم پیالہ	پیالہ	پیالہ	نانی	نانی	نانی
اضلاعوں	ضلع	ضلع	مرکوز	مرکز	مرکز
وضاحتوں	وضاحت	وضاحت	شخصیت	شخص	شخص
چشمکی	چشم	چشم	قاروا	قرو	قرو