Template Based Affix Stemmer for a Morphologically Rich Language

Sajjad Khan[1], Waqas Anwar[1,2], Usama Bajwa[1] and Xuan Wang[2]
[1]Department of Computer Science, COMSATS Institute of Information Technology, Pakistan
[2]Harbin Institute of Technology Shenzhen Graduate School, China

Abstract: *Word stemming is one of the most significant factors that affect the performance of a Natural Language Processing (NLP) application such as Information Retrieval (IR) system, part of speech tagging, machine translation system and syntactic parsing. Urdu language raises several challenges to NLP largely due to its rich morphology. In Urdu language, stemming process is different as compared to that for other languages, as it not only depends on removing prefixes and suffixes but also on removing infixes. In this paper, we introduce a template based stemmer that eliminates all kinds of affixes i.e., prefixes, infixes and suffixes, depending on the morphological pattern of the word. The presented results are excellent and this stemmer can prove to be very affective for a morphologically rich language.*

Keywords: *IR, stemming, prefix, infix, suffix, exception lists.*

Received October 19, 2012; accepted January 20, 2013; published online April 23, 2014

1. Introduction

For many researchers Human Computer Interaction (HCI) is a significant field of interest. Natural Language Processing (NLP) is one of the core research areas that support HCI.

Urdu NLP is one of the new research areas being explored since recent past. Urdu is an Indo-Aryan language. The word "Urdu" itself comes from a Turkish word "ordu", meaning "tent" or "army". Therefore, Urdu is some times called "Lashkari Zaban" or "The Language of the Army". It is the national language of Pakistan and is one of the twenty-three official languages of India. It is written in Perso-Arabic script. The Urdu's vocabulary consists of several languages including Arabic, Turkish, Sanskrit and Farsi (Persian).

Urdu's script is right-to-left and form of a word's character is context sensitive, means the form of a character is dissimilar in a word because of the location of that character in the word (beginning, centre and on the ending).

Information Retrieval (IR) system is used to make easy access to stored information. It also, deals with the saving, representation and organization of information objects. Modules of an IR system consist of a group of information objects, a group of requests and a method to decide which information items are most possibly to meet the requirements of the requests. Principally, customers of information shape their information needs in the form of queries and forward them to the IR system. Depending upon a query, the system yields information items that are supposed to be related to the query. In other words, the objects that are common in both the query and the database are returned to the clients. This occurs when a match is present between the information items and the queries.

To get a match between these two elements, the items should be represented in some form, both in the documents as well as in the query. Variations of a word present in a document prevent an exact match between a query word and the relevant document words. To overcome this issue, the replacement of the words with their relevant stems should be performed.

The stem is that portion of a word, which is left after affixes truncation (prefixes, suffixes and infixes). A usual example of a stem can be the word "mix" which is the stem for the variants "mixture", "mixed" and "mixable". Stemmer is an algorithm that reduces the word to their stem/ root form. Similarly, the Urdu stemmer should stem the words حاضری (Presence), حاضرین (The audience), غیرحاضر (Absent) to Urdu stem word حاضر (Present).

Stemming is part of a complex process which consists of taking out the words from text and turning them into index terms in an IR system. Indexing is the process of selecting keywords for representing a document. The essential part of any IR system for text searching is the capability to correctly recognize the variant word forms that take place ranging from grammatical variations to alternative spellings of the words in the query of user. Normally such variants are covered by means of right-hand truncation or through stemming. The truncation is performed by the user, who eliminates letters from the right hand side of the word to make it look like a possible root; the search then retrieves all words that begin with this root, despite their endings. On the other hand, Stemming is

carried out automatically by decreasing every word having identical root to common form, usually by eliminating derivational and inflectional suffixes. Urdu is a morphologically complex language. It uses both types of morphologies, i.e., derivational and inflectional. Therefore, depending on the morphological complexity of a language, both derivational and inflectional morphologies can result in a vast numbers of variants for a single word. Accordingly, word form variations can have a strong impact on the efficiency of an IR system and on morphological analysis tools.

The stemmer is also applicable to other NLP applications needing morphological analysis, for example, document compression, spell checkers, text analysis, word frequency count studies, word parsing etc., after analyzing different stemmers, we have noticed that most of the stemmers are based on the truncation of prefixes and suffixes to get stem without dealing with the infixes. In Urdu language, applying a light weight stemmer on a word having infix produces wrong stem by applying prefix and suffix stripping. Therefore, this study handles those words that contain infixes by using different templates.

The rest of the paper organization is as follows: In section 2 different stemming approaches are discussed, in section 3 characteristics of Urdu language are discussed, section 4 discusses the proposed urdu stemmer and finally the results of the stemmer are discussed in section 5.

2. Related Work

There are different approaches used for stemming. Some of the approaches are explained below:

- *Table Lookup Method* [16]: Is also, known as brute force method, where every word and its respective stem are stored in table. The stemmer finds the stem of the input word in the respective stem table. This process is very fast, but it has severe disadvantage i.e., large memory space required for words and their stems and the difficulties in creating such tables. This kind of stemming algorithm might not be practical.
- *Suffix Stripping Algorithms*: Do not just depend on a lookup table that contains inflected forms and their stem form. As an alternative, a list can also be maintained that contains "rules" which provide a path of execution for the algorithm, given an input word, to get its stem form. For example, if the word ends in "ing", then eliminate the "ing". If the word ends in "es", remove the "es". Suffix stripping approach only removes the suffixes from a word and it is much simpler to maintain. But, here we have to face over-stemming and under-stemming. The over-stemming happens when words that are not morphological variants are conflated e.g., the words compile and compute are both stemmed to comp. The under-stemming arises when words that are definitely morphological variants are not conflated e.g., compile being stemmed to comp and compiling to compil.
- *Suffix Substitution*: Is the improved version of suffix stripping. A number of rules are developed for specific suffixes in which a suffix will replaces with some other suffix e.g., the suffix "ies" is replaced with "y". Suppose a word "families", with the help of suffix substitution we can change this word to "family".
- *Lemmatization Algorithm*: Is a different method of finding the stem of a word. This method first finds out the part of speech of a word and then on each part of speech, normalization rules are applied. According to these rules stemming is performed.
- *Stochastic Algorithm*: Uses probability to discover the root form of a word. The algorithm is trained to develop a probabilistic model. The model consists of complex linguistic rules. The word is inputted to the trained model and then by using its internal linguistic rules, the root form of a word is generated.
- *Hybrid Approaches*: Use a number of approaches for stemming e.g., using suffix-stripping technique along with suffix substitution technique.
- *Affix Approaches*: Remove prefix and suffix from a word to get a stem word e.g., "unnoted" is a word in which "un" is a prefix and "d" is a suffix.
- *N-gram Analysis*: Is another technique, which is used for stemming. The word "gram" is a unit of measurement of text that may refer to a single character or a single word. The series of two grams is called bigram and series of three grams is called trigram. This technique uses a table of bigram and trigram words to find the stem of a word.
- *Iteration and Longest Match Algorithms*: Are used for stemming purpose. In the iteration method there is an order for classes for suffixes. The last order class, which happens at the end of the word, contains suffixes. Iteration algorithms use recursive procedure. This approach removes the string in order class, starts from the endings and then shifts towards the beginning. In the longest match principle here ending of the word is matched with the list of suffixes. If more than one ending offers a match then the longest match should be removed e.g., if there are two suffixes "ies" and "es" will be there then "ies" will be removed [16].

Lovins [8] published the first English stemmer and used about 260 rules for stemming the English language. The author suggested a stemmer consisting of two stages. The first stage removes the maximum possible ending which matches one on a redefined suffix list. The spelling exceptions are covered in the second stage.

Porter [12] developed a stemmer based on the truncation of suffixes, by means of list of suffixes and some conditions placed to recognize the suffix to be detached and generating a valid stem. Porter stemmer performs stemming process in five steps. The Inflectional suffixes are handled in the first step, derivational suffixes are handled through the next three steps and the final step is the recoding step. Porter reduced Lovin's rules to 60.

Different stemmers have also been developed for Arabic language. The Arabic stemmer called a superior root-based stemmer, developed by Khoja [7], truncates prefixes, suffixes and infixes and then uses patterns for matching to pull out the roots. The algorithm has to face many problems particularly with nouns. The study [17] created a stemmer, which is applied on classical Arabic in Quran to produce stems. For each Surah, this stemmer generates list of words. These words are checked in stop word list, if they don't exist in this list then corresponding prefixes and suffixes are removed from these words.

The study [15] proposed the Educated Text Stemmer (ETS). It is a simple, dictionary free and efficient stemmer that decreases stemming errors and has lesser storage and time requirements.

Bon was the first stemmer developed for Persian language [16]. Bon is an iterative longest matching stemmer. The iterative longest matching stemmer truncates the longest possible morpheme from a word according to a set of rules. This procedure is repeated until no more characters can be eliminated. Another study [9] proposed a Farsi stemmer that works without dictionary. This stemmer first removes the verb and noun suffixes from a word. After that it starts truncating the prefixes from that word.

Till date only one stemmer i.e., assas-Band has been developed for Urdu language [1]. This stemmer extracts the stem/ root word of only Urdu words and not of borrowed words i.e., words from Arabic, Persian and English words. This algorithm removes the prefix and suffix from a word and returns the stem word.

3. Urdu Language

The word "Urdu" is a derived word from Turkish language and it means a horde. Urdu, an Indo-European language of the Indo Aryan family, is spoken in Pakistan and India. It is the national language of Pakistan. Hindi and Urdu languages are similar to each other. Both the languages have originated from the dialect of Delhi region and other than the minute details these languages share their morphology. As Hindi has accepted many words from Sansikrit, Urdu has taken a large number of words from Arabic and Persian [4]. Urdu also borrowed vocabulary from English, Portuguese and Turkish.

There are two kinds of gender (masculine/ feminine) for Nouns in Urdu Language, two kinds of numbers (singular/ plural) and three cases (vocative, direct and oblique). All nouns in Urdu, when used within a sentence, will be inflected for number and case. Suffix identifies gender on verbs and adjectives (e.g., paagal → paagalpan, "madness", ghabraaanaa→ ghabraahat, "anxiety") and common suffixes can be used to drive nouns from other words. These forms are masculine and feminine nouns [3]. One notable point is that the borrowed Arabic and Persian plural forms of the noun are never inflected in Urdu.

Verb agreement is shown with subject or with direct object at a particular instance [4]. Verbs in Urdu language have two non-finite forms, root and infinitive. The infinitives comprise of a verbal stem and a suffix. The stem may itself comprise of a verbal root and suffix. (e.g., aanaa "to move", jaanaa "to go"). In this example /aa-/ and /jaa-/ are the root and /naa-/ is suffix. "From grammatical point of view, the infinitive forms of all verbs are marked masculine. They neither occur in the plural, nor in the vocative. There are many verbal forms of stem having different endings added to them to form the patterns of different verb forms" [10].

In Urdu, Subjunctive is the finite verbal form, which conveys the weak conjectures on the Urdu part of speech. Another two verbal forms that may be finite or non-finite are the perfective and imperfective participles. The imperfective particle ends with -taa, -tee, -tii, -tiin. In case of perfective particle ends with e.g., -aa, -ee, -ii, -iin. But, the situation is different in case of a verbal stem that ends in a vowel, where we should add a/y/ before masculine singular ending [10, 11]. The future verb forms in Urdu cannot be decided from the verb stems rather it is decided from subjunctive forms. The endings for the future forms are (e.g., -gaa, -gee, -gii). The tense and aspect are frequently shown through the use of irregular auxiliary verbs. Common use of semi auxiliary elements also gives various semantic connotations [10].

Although, a lot of research and study has been carried out about the history and theory of linguistics for Urdu language, but still some grammar rules remain controversial, for instance its possession of three or more than three cases. In Urdu post positions nouns take place after noun phrase head, which is an absolute contrast to English language where a variety of elements occur between preposition and the governed noun. This process has helped to reach a conclusion that Urdu thus has a lot of diversity of cases [10] e.g., genitive, accusative etc., the use of the form ka (e.g., larkay ka kam, "The work of the boy").

Urdu is written using alphabets that are derived from Persian alphabets which are derivatives of Arabic alphabets. Like Persian and Arabic, it is also read from right to left. While speaking, Urdu is quite similar to Hindi but is absolutely different in writing where as its written form is more similar to Persian and Pashto. Urdu differs from Arabic in writing

because it uses more complicated and convoluted Nastaliq script, mostly used for Urdu orthography and Arabic leads to follow a more modern naskh. For Urdu orthography, Perso-Arabic script is used extensively in Nastaliq style where it takes over its two-dimensional nature where characters are written from right to left and the digits are written from left to right [5]. It is a specific writing system where the letters change their glyph shape with respect to their next letters and this particular feature of Nastaliq language is called context-sensitive. This Nastaliq writing system for Urdu comprises diagonal, non-monotonic, cursive, context sensitive writing system with a noteworthy number of symbols, dots and other diacritics (Aerab). Two styles of writing i.e., Naskh and Taleeq were combined to formulate this spatially concise Nastaliq writing style [5].

وہ گلی نمبر ۵،مکان نمبر۲۷ میں رہتا ہے("He is residing in street number 5, house number 27").

4. Proposed Methodology

Our proposed stemmer is based on a light weight stemmer and word-pattern matching.

4.1. Light Weight Stemmer

Light weight stemming finds the representative indexing form of a word by the application of truncation of affixes [2]. The core objective of light weight stemmer is to preserve the word meaning intact and so increases the retrieval performance of an IR system. An Urdu word can have a more complex form if all affixes are attached to its root. By removing all the affixes from a word we will obtain the stemmed word which is not the root but the basic word without any affixes and so we sustain the meaning of the word and increase the search effectiveness.

Various lists are developed that help in finding the stem of an Urdu word. Such stemmer is rule based and works by truncating all possible affixes from a word but a main problem in this light stemming is that in many cases there is ambiguity i.e., a particular string of letters may or may not play a function of affix. We have introduced a method for detecting such ambiguities which finds if a specific sequence is an affix or is part of the original word. For this purpose Global Prefix Exceptional List (GPEL) and Global Suffix Exceptional List (GSEL) are developed which are discussed in the coming sections.

Our developed stemmer extracts the stem by removing the maximum length of prefixes and suffixes from a word. For this purpose we have sorted the prefix and suffix lists in descending order. The algorithm of light weight stemmer is given below:

Algorithm 1:

Step 1: *Input Word for stemming Word=(Prefix)-stem (Suffix)*
Step 2: *Open stop word list*
If Word exists in Stop word list
Return to Step1
Else go to Step3
Step 3: *Open GPEL*
If Word exist in GPEL
Then go to Step5
Else go to Steo4
Step 4: *OPEN Urdu prefixes file*
Read prefix one by one from the file until EOF reached
If match occurred
Then remove prefix from word
Word=stem-(Suffix)
Step 5: *Open GSEL*
If Word exist in GSEL
Then Word=stem
Go to Step8
Else go to Step6
Step 6: *Open Urdu suffixes file*
Read suffix one by one from the file Until EOF reached
If match occurred
Then remove suffix from word
Word=stem
Go to Step7
Else go to Step8
Step 7: *Open Characters-Add List (CAL) file*
If Word exists in CAL file
Then add the respective Character to the word
Word=stem (normalized)
Else
Word=stem
Step 8: *End*

- *Removing Stop Words*: To find the stem of any Urdu word, first the word is checked that whether if it is a stop word or not. If it is, then there will be no processing on that word and stemmer will take next word. For this purpose a list is maintained that contains Urdu stop words e.g., کا،کی،سے،لے،لی،نے،جو،کے. Some more examples of Urdu stop words can be seen in Table 1.

Table 1. Urdu stop word list sample.

پر	سے	کو	میں
تک	پہ	اوپر	نیچے
ساتھ	نزدیک	وہ	اور
بلکہ	یا	کے	کا
کی	نہ	پس	نہیں

- *GPEL*: There are some words that appear to contain a prefix, in fact it is not a prefix but it is the part of that word e.g., in the word "نانا", this word contain a prefix "نا", if it is removed then it produced a word "نا", which is incorrect from stemming point of view. Therefore, such type of words must need to be identified in advance and will be treated as an exceptional case. For our stemmer an exceptional list for prefixes is created called GPEL. Some examples of GPEL of Urdu words can be seen in Table 2.

Table 2. GPEL of Urdu words.

قبول	کثیر	وہم	عوام
ارزاں	شہرت	انسان	عیاش
ازل	اعتماد	عکس	عید
درج	عظیم	افضل	اتفاق
ادب	عمل	فقیر	اجرت

- *Removing Prefixes*: Prefix is that morpheme which is attached to the start of a word. The prefix may consist of only a single character, two or more than two characters and some times a word. A list of 180 prefixes is maintained for this purpose. Some examples of Urdu prefix words can be seen in Table 3.

Table 3. Urdu prefix list sample.

ا	پست	بلند	بے
بد	با	بر	پا
ان	ادھ	بلا	پاس
زبر	در	خار	تا
خوش	شب	سرد	تن

- *GSEL*: In some cases, a suffix is removed from a word and actually that suffix is a part of a stem word which should not be detached. An erroneous result will be produced due to irrelevant truncation of suffix e.g., in the word "پانی", when the suffix "نی" is removed then it produces stem "**پا**", which is incorrect. Therefore, such type of words should be treated as an exceptional case. Some examples of GSEL of Urdu words can be seen in Table 4.

Table 4. GSEL of urdu words.

لیاقت	فوت	سیکنڈ	صفت
ساقی	غنچۂ	کھال	خلا
دستیاب	کنوار	زیست	ضرورت
ڈھکن	اذیت	مرد	خون
بندوبست	ذات	عالم	مجبور

- *Removing Suffixes*: The suffix is that morpheme that is added to the end of a word. The suffix may consist of a character, more than a single character or a complete word. A list of suffixes, consist of 750 for Urdu text are collected after consulting relevant literature. Some examples of Urdu Suffix List Sample can be seen in Table 5.

Table 5. Urdu suffix list sample.

ات	باش	ترین	رس
آباد	بندی	خانہ	رو
ار	بوس	خوار	ریز
آزار	پسند	دار	زن
انداز	پناہ	دوز	کار

- *Normalizing the Stem*: When affix stripping algorithm is applied on Urdu words, then some time we get an incomplete word, e.g., after stripping affixes from a word "بےبندگی", we get stem "بند", which is incorrect stem. Therefore, this word should be added with the character **"ہ"** to form correct stem form i.e., "بندہ". For this reason five types of list are maintained for five characters; **(**ا، ہ، ت، ی، ن**)** called Characters-Adds List (CAL). Some examples of CAL can be seen in Table 6.

Table 6. CAL.

Add Character (o)
اثاث
منصوب
احاط
ادار
معاوض

Our proposed algorithm first checks the entered word whether if it is a stop word or not. If it is then no processing will be done on that word and next word will be entered. When a word is not a stop word then the word is checked in GPEL, if it exists then it means that the word has prefix(s) but should not be removed from the word because they are the part of stem. On the other hand, if it does not exist in GPEL then it means it has some prefix(s), thus, prefix rules are applied to remove the maximum prefix from the word.

The word is then checked in GSEL, if it exists then it is marked as the stem. But, if it does not exist in the list then it means this word has some suffix(s). Therefore, suffix rules are applied to remove the maximum suffix from the word.

To normalize the word form, the word is checked in five different lists maintained for **(**ا، ہ، ت، ی، ن**)** characters i.e., CAL. When a word is found in any of the five lists then respective character is added to produce a normal word form. After this the word is finally marked as a stem.

We cannot get stem word of an Urdu word by only stripping off prefixes and (or) suffixes e.g., عالم (religious scholars), تصاویر (pictures), طلبا (seeker). These words contain infixes and large amount of such type of words are present in Urdu. Thus, a light weight stemmer cannot handle words having infixes. For this purpose different templates are developed.

4.2. Template Based Matching

We have grouped the patterns based on the word's length. After that, rules of that pattern are applied on Urdu word to form stem word.

4.2.1. Urdu Words of Length Four Characters

- *Pattern "فعال"*: We will compare the word with patterns according to its length by using some conditions to find the infix in that word. Suppose we have a word "حکام" that has length 4 characters. To search the pattern for such word, we will set the conditions like:
  - Length of the word is 4.
  - 3rd character of Urdu word="ا".

  The above conditions match only the pattern "فعال". Now, by removing the 3rd character from the word, we will get the stem word حکم. Thus, various rules/ conditions are set for stemming.

- *Pattern "فعول"*:
  If 3rd character of Urdu word = "و"
  Then, removing the 3rd character form the word.
  Stem=First character +2nd character + 4th character
  Example: Word= بروج, Stem= برج.
  In some cases, after removing "و" character, we get a word in which two successive characters are same. Thus, in that case the last duplicated character should be removed. Example: word= خطوط after applying rule, we get its stem= خط.

4.2.2. Urdu Words of Length Five Characters

For five word length, the following patterns are proposed:

- *Pattern "افعال"*:
  If 1st and 4th characters of Urdu word = "ا"
  Then, remove these characters from the word.
  Stem=2nd character+3rd character+5th character
  Example: Word= ابدان, Stem= بدن.
- *Pattern "مفاعل"*:
  If 1st character of Urdu word = "م" and 3rd character of Urdu word= "ا"
  Then, remove the character at 3rd location from the word.
  Stem=1st character+2nd character+4th character+5th character
  Example: Word= مجالس, Stem=مجلس.
- *Pattern "افاعل"*:
  If 2nd character of Urdu word = "و" and 3rd character of Urdu word= "ا"
  Then, Stem=1st character+3rd character+4th character +5th character
  Example: Word=سواحل, Stem= ساحل.

  Some time we get an incomplete stem word, thus, normalization is required.

  If 2nd character of Urdu word is not "و" and 3rd character of Urdu word= "ا"
  Then, Stem=1st character+2nd character+4th character +5th character.
  Example: Word=سلاسل, Stem= سلسلہ.
  It is worth to mention that after applying rule on word "سلاسل", it gives "سلسل" as a stem, which is not a complete stem word. Thus, this word is searched in CAL and after adding character "ہ" to this word then it forms a correct stem word i.e., سلسلہ. Other patterns used for length five characters are "فعالی", "فعلان", "افعلہ" etc.

4.2.3. Urdu Words of Length Six Characters

- *Pattern "فعالیل"*:
  If 3rd character of Urdu word="ا" and 2nd character of Urdu word="و" and 5th character of the word="ی".
  Then, Stem=1st character +"ا" +4th character+" و+6th character
  Example: Word=خواتین, Stem=خاتون.
  Other patterns used for length six characters are "تفاعیل"etc.

4.2.4. Urdu Words having Hamza (ء) at the End

For the words having Hamza at the end, the following rules are developed:

Urdu Word having Four Character Lengths:
If 3rd character of Urdu word = "ا" and last character of Urdu word= "ء"
Then, Stem=1st character+2nd character+"ی"+3rd character.
Example: Word=ادباء, Stem=ادیب.

Urdu Word having Five Character Lengths:
If 1st and 5th characters of Urdu word= "ا" and last character of Urdu word= "ء"
Then, Stem=2nd character+3rd character+4th character
Example: Word=اذکیاء, Stem=ذکی, Word=انبیاء
Stem=نبی.

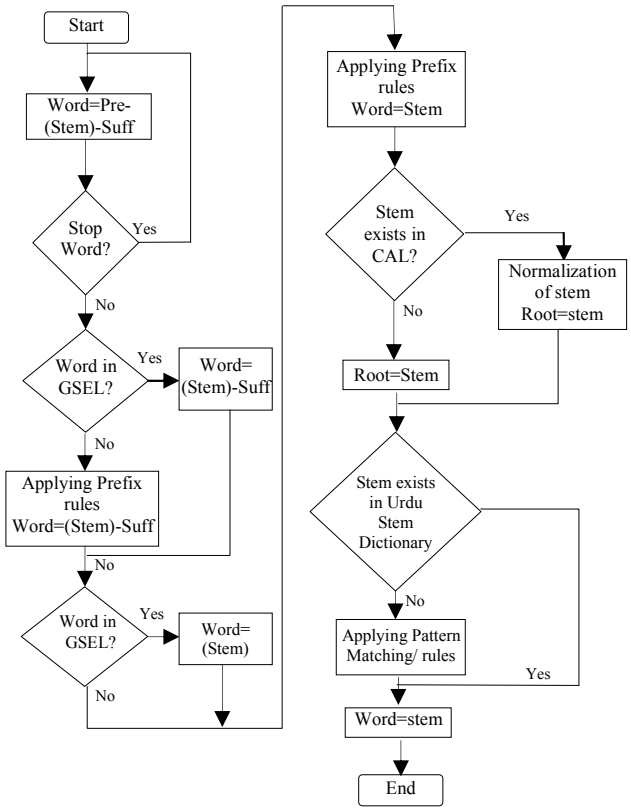

Figure 1. Flow chart of the urdu affix stemmer.

The flow chart of the Urdu Affix stemmer is shown in Figure 1. The algorithm of the Affix Urdu stemmer (Improved) is as below:

Algorithm 2:

Step 1: *Input Word for stemming*
Word=(Prefix)-stem-(Suffix)
Step 2: *Open Stop word list*
If Word exists in Stop word list
Return to step 1
Else go to step 3
Step 3: *Open GPEL*
If Word exist in GPEL
Then go to step 5
Else go to step 4
Step 4: *Open Urdu prefixes file*
Read prefix one by one from the file until EOF reached
If match occurred
Then remove prefix from word
Word=stem-(Suffix)
Step 5: *Open GSEL*
If Word exist in GSEL
Then Word=stem
Go to step 11
Else go to step 6
Step 6: *Open Urdu suffixes file*
Read suffix one by one from the file Until EOF reached
If match occurred

```
            Then remove suffix from word
            Word=stem
           Go to step 7
         Else go to step I
Step 7: Open Characters-Add List (CAL) file
          If Word exists in CAL file
             Then add the respective character to the word
             Word=stem (normalized)
          Else
            Go to step 9
Step 8: If step G is performed due to step 10
          Go to step 11
        Else
          Go to step 9
Step 9: Open Urdu stem dictionary
        Check the Root in dictionary
             If root exists in dictionary
               Then Word=stem and go to step 11
            Else
              Go to step 10
Step 10: Apply word patterns and rules:
        Find the length of word.
        Desired word's pattern is executed on the basis of
          word length.
        The appropriate rule of the desired word's pattern is
          applied on the word.
        Go to step 7
Step 11: End
```

The algorithm for section 4.2.2 to 4.2.4 is given below:

Algorithm 3:

```
Count Length
# Length of word (L)
L= Length
For (int i=0; i<=L; i++)
{
  If (length==4)
  {
     If 3rd character of Urdu word="ا" or "و"
    Then Stem=First character+2nd character+4th character
  }
      If (length==5)
   {
      If 1st and 4th characters of Urdu word="ا"
      Then Stem=2nd character+3rd character+5th character
      If 1st character of Urdu word="م"
      And 3rd character of Urdu word="ا"
      Then Stem=1st character+2nd character+
      4th character+5th character
      If 2nd character of Urdu word="و"
      And 3rd character of Urdu word="ا"
      Then Stem=1st character+3rd character+
     4th character+5th character
      If 3rd character of Urdu word="ا"
      And 4th character of Urdu word="ء"
      Then Stem=1st character +2nd character
      +"ی"+3rd character
   }
    If (length==6)
    {
       If 3rd character of Urdu word="ا"
      And 2nd character of Urdu word="و"
      And 5th character of the word= "ی"
      Then Stem=1st character+"ا"+4th character+
      "و"+6th character
       If 1st and 5th characters of Urdu word = "ا"
       And 6th character of Urdu word="ء"
       Then Stem=2nd character+3rd character
      +4th character
    }
  }
```

5. Experimental Results

We evaluated our Urdu stemmer on three corpora i.e., corpus-1 (9200 words), corpus-2 (27000 words) and corpus-3 (30000 words). These corpora include data in the form of verbs, nouns, adjectives, punctuations, numbers, special symbols etc.

- *Light Stemmer Results*: When the corpus-1 is fed to the stemmer, then in pre-processing step, the stemmer removes all stop words, numbers and punctuation marks. Thus, after pre-processing steps, there are 4268 words left. Stemming is performed on 4268 words that produce 39.4% precision, 71.1% recall and 50.70% F1-Measure. The low precision is due to occurrence of various issues in stemming Urdu words e.g., Compounding, Tokenization, Transliteration and Infixation [6].

  The same stemmer is applied on corpus-2 that produces 18378 words after pre-processing. It gives 49% precision, 78.6% recall and 60.36% F1-measure. Now, corpuse-3 is fed to the stemmer that produces 19351 words after pre-processing. Our light weight stemmer produces precision 73.55%, recall 90.53% and 81.16% F1-measure. The summary of the three corpora after applying stemmer is given in Tables 7 and 8.

Table 7. Shows the summary of the three corpora and light weight stemmer results.

Characteristics	Corpus-1	Corpus-2	Corpus-3
Words After Pre-Processing	4268	18378	19351
Already Stemmed Words	2233	10674	12428
Words to be Stemmed	2035	7704	6923
Correct Stemmed Words	802	3775	5092
Results			
Precision	39.4%	49%	73.55%
Recall	71.1%	78.6%	90.53%
F1-Measure	50.70%	60.36%	81.16%

- *Proposed Affix Stemmer Results*: When the proposed stemmer is applied on corpus-1, it produces 66.53% precision, 84% recall and 74.25% F1-measure. This stemmer is also applied on two more corpora i.e., corpus-2 and corpus-3. It gives 76.6% precision, 91.94% recall and 83.57% F1-measure for corpus 2 and for corpus-3 (training data) it produces 89.05% precision, 96.08% recall and 92.49 %F1-measure.

Table 8. Shows the summary of the three Corpora and affix stemmer Results.

Characteristics	Corpus-1	Corpus-2	Corpus-3
Words After Pre-Processing	4268	18378	19351
Already stemmed Words	2233	10674	12428
Words to be Stemmed	2035	7704	6923
Correct Stemmed Words	1354	5902	6165
Results			
Precision	66.53%	76.6%	89.05%
Recall	84%	91.94%	96.08%
F1-Measure	74.25%	83.57%	92.49%

6. Conclusions

Morphologically Urdu is a complex language. There exist a number of variants for a single word in this language. Urdu is rich in both inflectional and derivational morphologies.

The light weight stemmer removed prefixes and suffixes from a word to get stem word. After applying this stemmer on a corpus, it was noted that only removing prefixes and suffixes cannot produce a stem of that word but there is a lot of words in Urdu that contain infixes too. Therefore, this light weight stemmer was improved and the proposed stemmer used different word patterns and rules for those words that contain infixes. The proposed stemmer (improved stemmer) was run on the training data. This stemmer gave 89.05% Precession, 96.08% Recall and 92.49% F1-Measure.

The stemmer increased recall at the cost of decreased precision. This research proves that maximum matching affix approach is more suitable for developing a stemmer for Urdu language.

References

[1] Akram Q., Naseer A., and Hussain S., "Assas-Band, an Affix-Exception-List Based Urdu Stemmer," *in Proceedings of the 7th Workshop on Asian Language Resources*, Singapore, pp. 40-47, 2009.

[2] Al-Sughaiyer I. and Al-Kharashi I., "Arabic Morphological Analysis Techniques: A Comprehensive Survey," *the Journal of the American Society for Information Science and Technology*, vol. 55, no. 3, pp. 189-213, 2004.

[3] Anwar W., Xuan W., and Xiao-long W., "A Survey of Automatic Urdu Language Processing" *in Proceedings of the International Conference on Machine Learning and Cybernetics*, Dalian, China, pp. 4489-4494, 2006.

[4] Hardie A., "Developing a Tagset for Automated Part-of-Speech Tagging in Urdu," *in Proceeding of the Corpus Linguistics conference, Department of Linguistics*, Lancaster University, UK, pp. 298-307, 2003.

[5] Hussain S., "Complexity of Asian Scripts: A Case Study of Nafees Nasta'leeq," *Invited Talk at SCALLA*, Kathmandu, 2004.

[6] Khan S., Anwar W., and Bajwa U., "Challenges in Developing a Rule based Urdu Stemmer," *in Proceedings of the 2nd Workshop on South and Southeast Asian Natural Language Processing*, Thailand, pp. 46-51, 2011.

[7] Khoja S., *Stemming Arabic Text*, Computing Department, Lancaster University, UK, 1999.

[8] Lovins B., "Development of a Stemming Algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, no. 1, pp. 22-31, 1968.

[9] Mokhtaripour A. and Jahanpour S., "Introduction to a New Farsi Stemmer," *in Proceedings of the 15th ACM International Conference on Information and Knowledge management*, Virginia, USA, pp. 826-827, 2006.

[10] Naim M., *Introductory Urdu*, South Asian Language and Area Center, University of Chicago, USA, 1999.

[11] Platts T., *A Grammar of the Hindustani or Urdu Language*, Crosby Lockwood and sons, London, UK, 1909.

[12] Porter F., "An Algorithm for Suffix Stripping," *Program*, vol. 14, no. 3, pp. 130-137, 1980.

[13] Ramanathan A. and Rao D., "A Lightweight Stemmer for Hindi," *in Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics for Computational Linguistics for South Asian Languages*, Toulouse, France, pp. 42-48, 2003.

[14] Rehman Z. and Anwar W., "Hybrid Approach for Urdu Sentence Boundary Disambiguation," *the International Arab Journal of Information Technology*, vol. 9, no. 3, pp. 250-255, 2012.

[15] Tamah E. and Lin J., "Towards an Error-Free Arabic Stemming," *in Proceeding of the 2nd ACM workshop on Improving Non English Web Searching*, *inews*, California, USA, pp. 9-16, 2008.

[16] Tashakori M., Meybodi M., and Oroumchian F., "Bon: First Persian Stemmer," *in Proceedings of the 1st EurAsian Conference on Information and Communication Technology*, Shiraz, Iran, pp. 487-494, 2002.

[17] Thabet N., "Stemming the Qur'an," *in Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*, Geneva, Switzerland, pp. 85-88, 2004.

Sajjad Khan is a Senior Instructor in computer science at Forest Department, Khyber Pakhtunkhwa, Pakistan since 2002. He got his Ms degree in computer science from COMSATS in 2011. He did Ms degree in computer science from University of Peshawar, Pakistan in 2001. His area of interest is natural language processing and text mining.

Waqas Anwar is currently working at COMSATS Institute of Information Technology, Pakistan as an Assistant Professor since 2012. He got his PhD degree in computer application technology from Harbin Institute of Technology, PR China in 2008. He did Ms degree in computer science from Hamdard University, Pakistan in 2001. He is an active researcher and his areas of interest are natural language processing and computational intelligence.

Usama Bajwa is currently serving as an Assistant Professor at Computer Science Department, COMSATS Institute of Information Technology, Pakistan. Previously he has served as a team lead in an ICT R and D project funded by Ministry of IT and Telecom, Pakistan and as a visiting researcher at the Medical Imaging Lab, University of Glamorgan, UK. He did his Ms degree in computer engineering from Center for Advance Studies in Engineering, Pakistan in 2006 and Masters in computer science from Hamdard University, Pakistan in 2001. He is currently pursuing his PhD degree at Center for Advance Studies in Engineering, Pakistan. His research interests include biometrics, pattern analysis and natural language processing.

Xuan Wang received his PhD degree in computer science and technology from Harbin Institute of Technology in 1997. Currently, he is a professor of computer science at Harbin Institute of Technology Shenzhen Graduate School. His research interest includes artificial intelligence, computational linguistics and bioinformatics.