J Inf Process Syst, Vol.11, No.3, pp.450~464, September 2015
http://dx.doi.org/10.3745/JIPS.04.0020

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Comparative Study of Various Persian Stemmers in the Field of Information Retrieval

Fatemeh Momenipour Moghadam* and MohammadReza Keyvanpour**

Abstract
In linguistics, stemming is the operation of reducing words to their more general form, which is called the 'stem'. Stemming is an important step in information retrieval systems, natural language processing, and text mining. Information retrieval systems are evaluated by metrics like precision and recall and the fundamental superiority of an information retrieval system over another one is measured by them. Stemmers decrease the indexed file, increase the speed of information retrieval systems, and improve the performance of these systems by boosting precision and recall. There are few Persian stemmers and most of them work based on morphological rules. In this paper we carefully study Persian stemmers, which are classified into three main classes: structural stemmers, lookup table stemmers, and statistical stemmers. We describe the algorithms of each class carefully and present the weaknesses and strengths of each Persian stemmer. We also propose some metrics to compare and evaluate each stemmer by them.

Keywords
Lookup Table Stemmer, Stemmer, Statistical Stemmer, Structural Stemmer

1. Introduction

In the Persian language, a word usually consists of a stem (root) and affixes. The main part of a word is its stem. The affixes are added to the stem to change the meaning or grammatical rule of the word [1]. In linguistics, stemming is the operation of reducing words to their stem [2,3]. In the other words, a stemmer is software [4] that reduces all kinds of the word forms to the same morphological root, which is called the 'stem' [5]. Stemmers are used in natural language processing and some other fields, such as information retrieval and web searches [4]. For example, in IR, stems are used instead of the words. Therefore, the performance of the system is improved [6] and it also decreases the size of the indexing files [7]. Furthermore, stemming is an important step in processing textual data and text mining [8].

Farsi is an Indo-European language. It is spoken and written in Iran and in some parts of Tajikistan and Afghanistan. Suffixes and prefixes are added to the words to change their meaning. Farsi is written from right to left, so prefixes are added to the right side of the word and suffixes are added to the left side of the word. Affixes are added to the nouns to modify possession, meaning, and plurality [9]. Farsi also has its own properties like not using accents (except in some cases) and polymorphism in writing.

Manuscript received June 12, 2014; accepted August 05, 2015.
Corresponding Author: Fatemeh Momenipour Moghadam (sonaymomeni@gmail.com)
* Dept. of Computer Engineering, Islamic Azad University, Qazvin Branch, Qazvin, Iran (sonaymomeni@gmail.com)
**Dept. of Computer Engineering, Alzahra University, Tehran, Iran (Keyvanpour@alzahra.ac.ir)

One of the effective ways for improving the performance of IR systems is using a stemmer to find the stem of search terms [10]. There are three approaches that are used in stemming: structural stemming, lookup table, and statistical methods.

Structural stemming depends on the morphological structure of the language. To find the stem of the word, it removes the suffixes and prefixes of the words if they exist. Most of the Persian stemmers use this method. In the lookup table approach, because all words and their stems are stored in a database we can find the stem of the word by searching the database.

This approach needs a large amount of space and also it needs to be updated if a new word is added. In the statistical approach, some methods are used based on statistical principles, which are produced from the corpus of the documents. The advantage of this approach is its independence of the morphological structures of the language [2]. In this paper, we study Persian stemmers carefully, find the weaknesses and strengths of each one and then classify them into three main classes: structural stemmers, lookup table stemmers, and statistical stemmers. The rest of the paper is organized as follows: in Section 2, we study each Persian stemmer carefully and classify them in three groups. In Section 3, we find some metrics and compare and evaluate each model by them, and finally, in Section 4, we present the conclusion of our work.

2. Classification of Persian Stemmers

Increasing the performance of the information retrieval system has been an important factor for developing many efficient systems. Using stemmed words instead of the real words, may lead to improving the recall and precision of information retrieval systems [3]. Stemming also decrease the size of index files. This is due to the fact that a stem can be used instead of various structures of the words.

There are few stemmers in the Persian language. As such, designing a stemmer for the Persian language is difficult, which is due to its complicated grammar rules and numerous exceptions. There are three main approaches for Persian stemming. First, there is the structural approach, which uses the morphological structure to find the stem of every word. The second approach is the lookup table, which saves each word and all related forms of them in structural forms in a database. This makes it possible for us to find the stem of the stored words by searching the database. The last approach is the statistical method, which uses statistical and machine learning techniques. Fig. 1 shows the classes and subclasses of each group.

2.1 Structural Stemmers

Structural approaches use morphological rules to find the stem of every word [10]. They can find the stem by removing the suffixes and prefixes. These kinds of stemmers usually find the correct stem, but sometimes they need to change the results to find the correct stem. These algorithms follow several rules. Most of them usually try to remove the longest substrings of the words according to structural stemming rules. They do this process until there is nothing more to remove. Most of the Persian stemmers use this approach.

2.1.1 Bon stemmer

Bon is the first Persian stemmer. It finds the longest substring of a word and removes them from the word. It repeats this process until it there are no more characters to remove. Bon uses the recording technique to transform AXC→AYC, where X is the input string and Y is the string that is transformed. A and C are for the context transformation. Bon uses one dictionary of Persian infinitives, one for 'mokassar' words and their singular form, and another dictionary for insensitive words [4]. Because of the numerous exceptions that apply to Persian words, Bon cannot find the correct stem even after all characters are removed through a set of rules [1]. Fig. 2 shows the stemming process in the Bon stemmer.

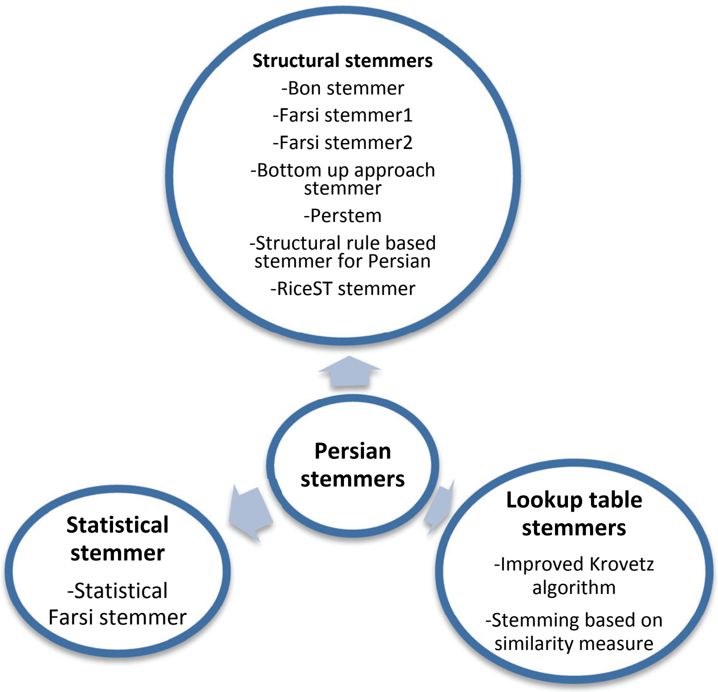

Structural stemmers
-Bon stemmer
-Farsi stemmer1
-Farsi stemmer2
-Bottom up approach stemmer
-Perstem
-Structural rule based stemmer for Persian
-RiceST stemmer
Persian stemmers
Statistical stemmer
-Statistical Farsi stemmer
Lookup table stemmers
-Improved Krovetz algorithm
-Stemming based on similarity measure

Fig. 1. Classification of Persian stemmers.

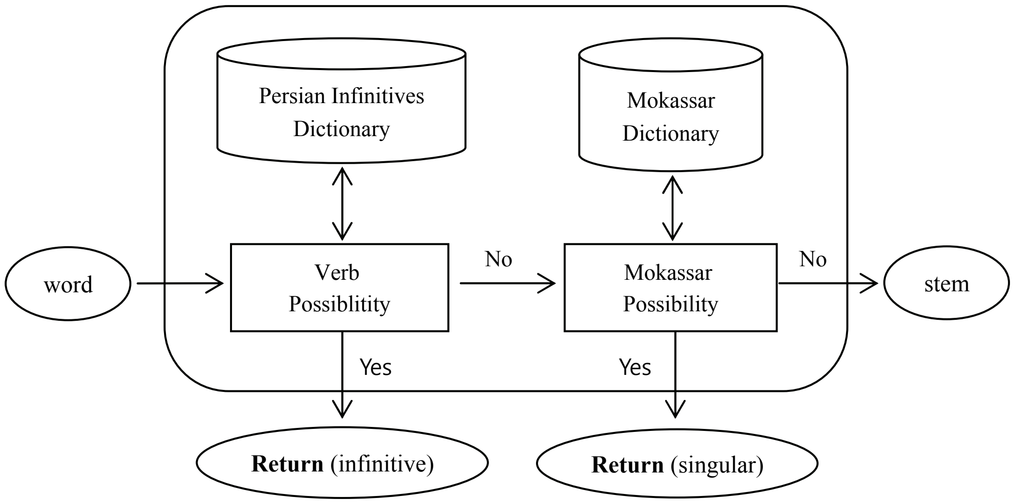


Fig. 2. Stemming in Bon stemmer.

Four hundred and fifty abstracts of Persian text, which were about computer science and called Persian computer abstract (PCA), were collected to test this stemmer. They used 32 queries to conduct experiments on this collection. Tashakori et al. who made this stemmer [4], claimed that this stemmer could increase recall by 40% on PCAs.

2.1.2 Farsi stemmer1

Taghva et al. [10] presented a structural Farsi stemmer in 2005. This stemmer finds the suffixes of a word according the list of Farsi suffixes, which was made by hand using Farsi grammar. If a word has multiple suffixes the stemmer selects the longest suffix [8,9].

The lists of suffixes were categorized into groups like verb-suffixes, plural-noun-suffixes, possessive-noun-suffixes, other-noun-suffixes, and other-suffixes. This categorization helps to remove the Persian of the verbs and suffixes of a noun. Suffixes are stacked where the word is a noun, in accordance with this pattern:

{Possessive-noun-suffix}{Plural-noun-suffix}{Other-noun-suffix}<Stem>

To specify the suffixes of the input words, the Farsi stemmer applies deterministic finite automata (DFA). The DFA begins at the end of input word and works until the third letter from the front.

To test this stemmer, a collection of 1,647 Farsi documents and 60 queries were used. This stemmer was tested in a vector-based information retrieval system. The test showed an 18% increase in the precision of the information retrieval system. This stemmer was tested on a small collection and so, the effect of the stemmer on larger collections is unknown [10].

2.1.3 Farsi stemmer2

Mokhtaripour and Jahanpour [11] studied the Farsi language in-depth and tried to achieve three goals that made the stemmer more accurate. These goals were as follows:

- Searching the affixes
- Finding the changes to the words after adding affixes
- Paying attention to the loan words.

Fig. 3 shows their main operations on the words in Farsi stemmer2:

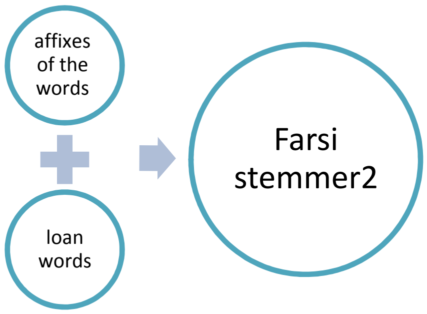


Fig. 3. Main work on words in Farsi stemmer2.

This stemmer is a rule-based one that has 10 phases. Every phase follows one or more specific grammar rules and after each rule, the length of the rest of the word shouldn't be less than two characters. This stemmer begins its work by finding the suffixes of the nouns and verbs and removing them. It then tries to remove the prefixes and considers the changes to the words in all of the removal steps.

The 10 removal phases are as follows:

- Removing "ی "(/_/) (indefinite article / possessive suffix)
- Removing the auxiliary suffix " ند " /_nd/
- Removing possessive and auxiliary suffixes
- Removing the possessive suffixes " ت" /t/ and " تان "
- Removing plural suffixes
- Removing comparative suffixes
- Removing other suffixes
- Removing " ن" /n/ (sign of infinitive)
- Removing special end letters
- Removing prefixes.

There are some Arabic words in Farsi. If the stemmer ensures that a word is Arabic, it applies some of the Arabic rules to find the stem of the word.

A set of 43,680 Farsi documents, which covered various subjects like sport, economics, policy, history, etc. [11], were used to test this stemmer. Twenty-five queries that were related to the documents were also prepared. A classic vector-based system was run without the stemmer in the indexer and the performance of the system was measured. Then, this system was run with the current stemmer and the precision and recall were measured. The results showed that the system has a 46% improvement rate if it used this stemmer [8]. The weakness of this work is that they used a small collection to evaluate the performance of the system and the effect of the stemmer on a bigger collections remains unknown.

2.1.4 A bottom up approach stemmer

Sharifloo and Shamsfard [2] presented a bottom up stemmer that is rule based. Their algorithm has three steps, which are shown in Fig. 4.

In Step 1, they defined which parts of the word were morphemes and which parts were not. In fact, they found the morphological information of all words. They also defined the cluster of each morpheme and the cores in the words. In Step 2, they defined the related rules for the cores that were defined in Step 1. For instance, "خور"(khor) is one core of the word "می خورم" (mi-xor-am: "I eat") and it is one of the members of the cluster "بن مضارع" (bone mozare: present root), "م" (am) is one of the members of the cluster "شناسه مضارع" (shenase mozare: present person identifier), and "می" is a member of the cluster "می" (mi) [9].

(می + بن مضارع + شناسه مضارع)

(Present person identifier + present root + mi)

The result of the second step was a list of cores that extracted all of the rules for the word. In the last step, anti-rule matching, they extracted the anti-rules from the anti-rules repository for every core of the core list. If a core is matched with the word morpheme, it removes it from the core list. As a result, only the correct stems of the words are included in the core list.

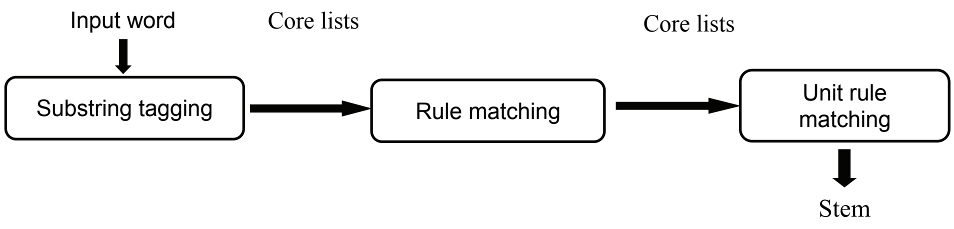


Fig. 4. Phases of a bottom up approach stemmer.

This stemmer was evaluated with a limited corpus of the newspaper, *Hamshahri*. Sharifloo and Shamsfard [2] began their test with 252 rules and 20 anti-rules. This algorithm had a 90.1% correct stemming rate.

2.1.5 A structural rule-based stemmer for Persian

Rahimtoroghi et al. [1] used heuristic rules based on the structure of Persian and its exceptions, and created 33 rules to stem the words. They were able to determine some suffix removal rules and used them to stem the words. These rules are shown below in Table 1.

Table 1. The rules of Farsi word suffixes

Length	Suffix for removal	Number of stemming rules based on the criteria
Words length>6	Suffix "ترین" in English "tarin"	2
	Suffix "گیری" in English "geeree"	2
	Suffix "آباد" in English "aabaad"	2
	Suffix "سازی" in English "saazee"	2
	Suffix "هایی" in English "haayee"	4
	Suffix "ریزی" in English "reezee"	4
	Suffix "بندی" in English "bandee"	2
Words length>5	Suffix "های" in English "haay"	2
	Suffix "شان" in English "shaan"	1
Words length>4	Suffix "ان"in English "aan"	1
	Suffix "ها" in English "haa"	1
	Suffix "ات"in English "aat"	1

A search engine called Lucene, was used to evaluate this stemmer. This search engine used a probabilistic model called Okapi BM25. This model applied the following formula to calculate the score of document *d* and query *q*:

$$\mathrm{RSV_d} = \sum_{t\in q} \log \frac{N}{\mathrm{dft}}.\frac{(K1+1)tf\ td}{k1\left((1-b)+b\times(\frac{Ld}{Lave})\right)\ \ +tf\ td} \tag{1}$$

RSV stands for retrieval status value. In this formula, N is the number of whole documents, df_t is the number of documents that contain t, tf_{td} is the number of occurrences of term t in document d, Ld is the length of the document d, and L_{avg} is the average length of the document. $K1$ is set to 1,000 and b is set to 0.75 in the Okapi formula.

Hamshahri collection was used to test this stemmer. The results showed a 4.78% growth rate in the precision of the information retrieval system. The size of the indexed file was decreased, so the speed of the system increased [1].

2.1.6 Perstem

Perstem, is a morphological analyzer [12]. It uses some regular expression replacements to separate morphemes from stem. Words are first searched for in a hash table. If the word exists in a table, it is the stem and no other work is conducted [11, 13]. If not, then some regular expression substitutions should be carried out. If the stemmer finds a prefix, it puts a +_ between the prefix and the stem and if it finds a suffix, it puts a _+ between the stem and the suffix. Then, the stemmer deletes all prefixes and suffixes, so the rest of the characters are the stem .This stemmer processes 15,000 words in a second on a desktop computer. Jadidinejad et al. [13] tested Perstem. First, they used the *Hamshahri* corpus and Indri search engine (this test is called "Baseline"), and then Perstem analyzed documents and queries in the *Hamshahri* corpus and created a new one. This was called Perstem. The experiments showed improvement in recall and precision [14].

2.1.7 RICeST Stemmer

In [12], linguistic knowledge and algorithms that support 10 suffixes and almost 2,000 exceptions were used to create the RICesT stemmer. There are two kinds of plural suffixes, "ha" and "an," which are used to turn singular nouns into plural nouns, like "ghazaha" and "derakhtan," and "at," "in," and "un," which are mostly used to make plural Arabic words like "sabzijat," "rohaniyun," and "sareghin." It is sometimes necessary to add a letter between words and suffixes [2] (i.e., "chaharpa"+"a" is changed to "chaharpayan") . A singular system produces the single form of the plural nouns by applying the following rules:

- Find the plural nouns and suffixes
- Remove plural suffixes
- Make the exceptions distinct
- Present the singular form of irregular plural nouns.

A singular stemmer or s-stemmer uses more than 11 rules. The different parts of the s-stemmer are shown in Fig. 5.

Lexical, structural, and syntactical processes are applied to find the stem of a plural noun. The processor that has pattern-based and language methods have four parts, which are as described below.

1. Syntactic analyzer: this finds the suffixes of a noun.
2. Structural lexical analyzer: if unknown words enter the system, this reduces the noun to a predetermined form.
3. A noun as a lexical unit: this is where the noun is known as a lexical unit.

4. The forth part of the RICeST stemmer is related to the special exceptions and tries to find the single form of the words as outputs.

To find noun lexical knowledge, the system finds a list of grammatical rules that is related to a noun. If there is a new noun, it will be added to the grammatical rules list. The plural nouns will be converted to singular forms via means of natural language processing. The s-stemmer has the following advantages: it can find different forms of a noun and it can automatically classify the text in a large file. Users can personalize the system according to their needs, and create a statistical report of their work. This stemmer was installed in the Regional Information Center for Science and Technology [15].

2.2 Statistical Stemmers

In the statistical method, some rules are formulated based on statistical principles [17] through the inference process and by use of a corpus. This approach doesn't require any linguistic knowledge. The stemmer can stem the words of new languages with little effort, and this is one of the advantages of this approach, especially for digital libraries that need to manage documents written in more than one language. In [18] a statistical Persian stemmer is presented and is described below.

2.2.1 Statistical Farsi stemmer

A statistical Persian stemmer has been presented by Nasiri et al. [17] based on the Bacchin algorithm. Bacchin used statistical methods to distinguish the structures, forms, and grammatical rules of the words in 2002 and completed the implementation of the algorithm in 2005 [18]. He used a complete set of a language words and divided each word into two parts with the first part as a prefix and the second part as a suffix. Each substring was known as the node of a graph, an edge between two nodes, and showed a word of the dataset. Then the links of a substring were analyzed by statistical methods. In the Bacchin algorithm, W shows a dataset of the words, U is used as substrings of the words, and Ω is the combination of two substrings that make a word that exists in a dataset. The formula is shown in (2) [18]:

$$\Omega = \{(x, y) \in U \times U: \ni Z \in W, Z=XY\} \quad (2)$$

Two members of Ω have a relation to each other, which is shown by $F(z)$, if and only if they were made of the same word like Z. $F(z)$ partitions Ω into equivalent classes, which are shown by $\Omega(z)$:

$$\Omega(z) = \bigcup_{i=1}^{n-1} wi \quad (3)$$

which are the various states of substrings of a word into two parts. Based on Bayesian rules [5]:

$$W(z) = \arg\max \Pr(w) = \arg\max \Pr(yi|xi) \Pr(xi) \quad (4)$$

$$\Pr(Xi) = \sum_{j=1}^{n} \Pr(xi|yi) \Pr(yi) \text{ for } i=1, 2, N \quad (5)$$

$$\Pr(Yi) = \sum_{i=1}^{n} pr(yi|xi)\, pr(xi) \text{ for } j=1, 2, N \quad (6)$$

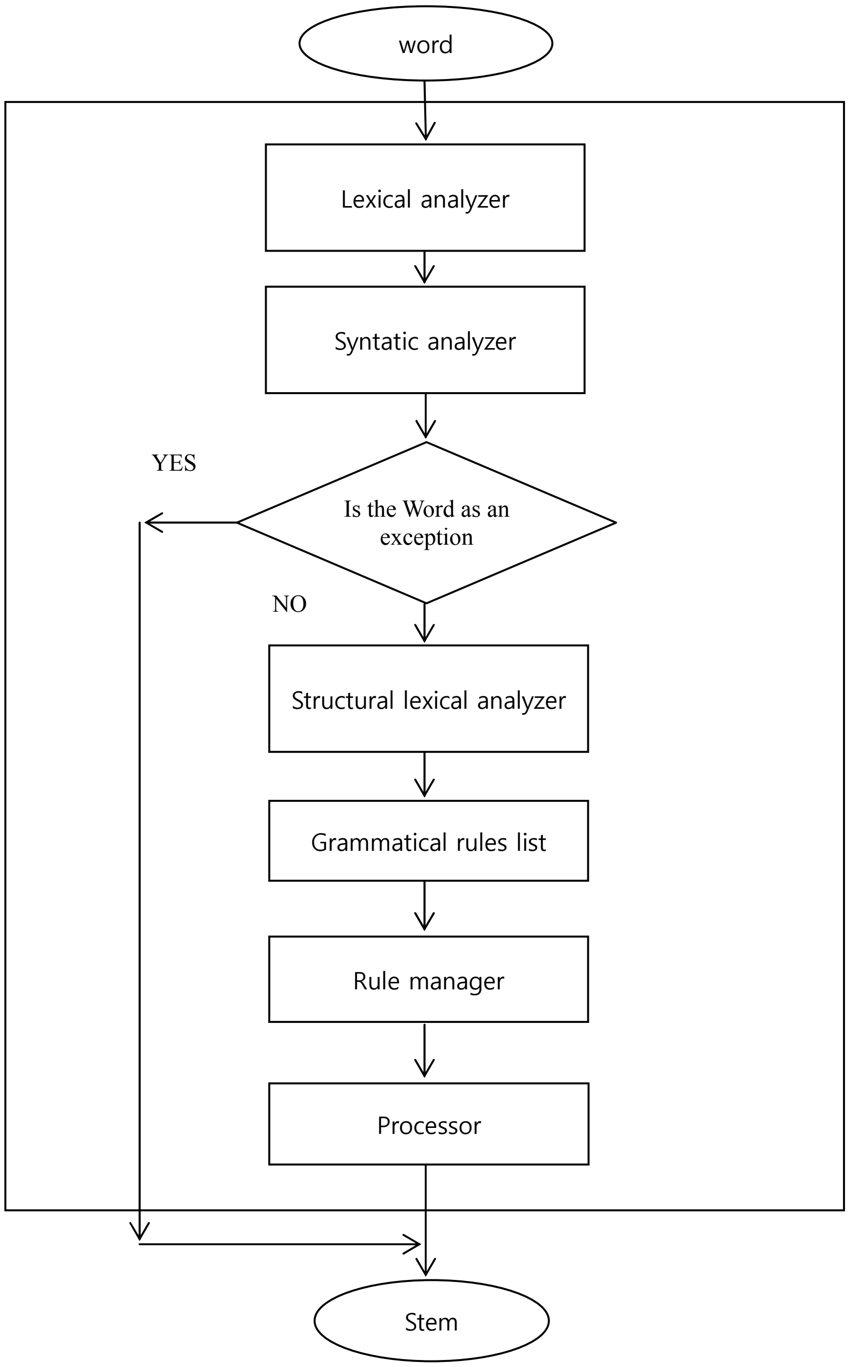


Fig. 5. Different parts of singular-stemmer system.

The main purpose of the algorithm is to estimate Pr(yi) for all substrings. Nasiri et al. [17] used the Bacchin algorithm and improved it for Persian stemming. In this stemmer, as shown in Fig. 6, the system finds the lists of the prefixes and suffixes of the words and assigns them weights in the learning phase. It then finds all the substrings of a word in the testing phase and finds the best stems by using the affix list. They used various states of words and found their best stems. They used 800 web documents of ISNA [19] web pages and 32 queries to test this Persian stemmer and because they couldn't find any search engine for the Farsi language, they used Lucece [20]. They claimed that their stemmer worked well. But, although this model was more complicated than structural approaches, it did not achieve better improvements than structural stemmers. The results showed that structural stemmers work better [1].

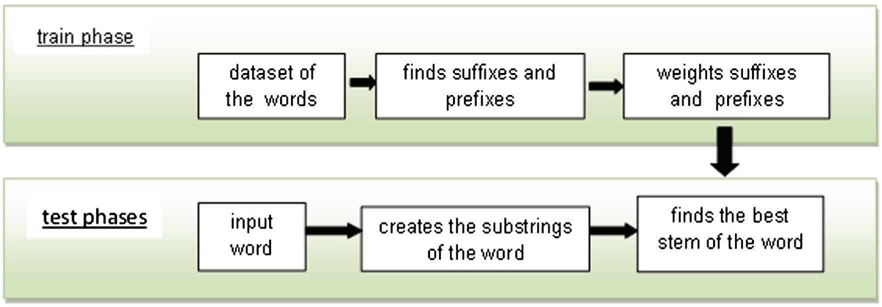

train phase
dataset of the words
finds suffixes and prefixes
weights suffixes and prefixes
test phases
input word
creates the substrings of the word
finds the best stem of the word

Fig. 6. The phases of learning process in statistical Farsi stemmer.

2.3 Lookup Table Approach

In this approach, the words and all related forms of them are saved in structural forms [2] in a database. This makes it possible for the stem to be found by searching in the database. This is a simple and accurate method, but it does have some problems, which are as follows: saving the database requires a large amount of space and this database need updating [9]. Hesamifard and Ghassem-Sani [21] proposed an algorithm based on the lookup table method, which is described below.

2.3.1 Improved Krovetz algorithm

Hesamifard and Ghassem-Sani [21] designed a stemmer based on the database's information. With this method, all of the words and their structure are saved in a database. To find the stem of a word, the input word has to be searched for in a database. If it exists in a database, the stem will be returned, but if the word does not exist in a database, the suffixes and prefixes have to be removed from both sides of the word and the rest of the word has to be searched for in a database again. This algorithm was designed for verbs. There are two different stems for Farsi verbs—past tense and present tense. Adding prefixes and suffixes to them creates all of the verbs of the Farsi language. The rules for making verbs are shown in Table 2. This algorithm does not work right in some states. So, a second version of it was made, called krovetz2. In the krovetz2 algorithm, the verb tenses (past or present) have to be determined first. This solved most of the issues. Krovetz algorithms were tested on 1,000 Farsi sentences. The verbs of the sentences were determined and tested by the krovetz stemmer. Both krovetz and krovetz2 received good results. In krovetz, 223 verbs were tested by the algorithm. This algorithm only made 13 mistakes and krovetz decreased this value to four mistakes. The results showed better results for the krovetz2 algorithm [21]. This algorithm has some weaknesses in that its database needs to be updated and its speed is low [9].

Table 2. Farsi verbs table

Composition	Verb
بن ماضی +{م،ی، یم ، ید، ند} **(past root+ {am,ei,iim,iid,and})**	Past simple
می + ماضی ساده **(mi+ simple past)**	Past continuous
بن ماضی + ه + {ام ، ای ، است ، ایم ، اید ، اند} **(past root+ eh+{am,ei,ast,eim,eid,and })**	Present perfect
بن ماضی + ه + بود+ {م ، ی ، یم ، ید ، ند} **(past root+ eh+{am,ei,ast,iim,iid,and })**	Past perfect
بن ماضی + ه + باش + {م ، ی ، یم ، ید ، ند} **(past root+ eh+baash+{ am,ii,im,id,and })**	Past implicit
می + بن مضارع + {م ، ی ، د ، یم ، ید ، ند} **(present root+{am, ii,ad,iim,iid,and})**	Present continuous
ب + بن مضارع + {م ، ی، د ، یم ، ید ، ند} **(Be+ present root+ {am, ii,ad,iim,iid,and})**	Present simple
ن + بن مضارع + {م ، ی ، د ، یم ، ید ، ند} **(Ne+ present root+ {am,ii,ad,iim,iid,and})**	Negative present simple
ب + بن مضارع + {""+یم ، ید} **(Be+ present root+ {“”+iim,iid})**	Imperative
ن + بن مضارع + {""+ یم ، ید } **(Na+ present root+ {“”+iim,iid})**	Negative imperative
ن +{ماضی ساده ، ماضی استمراری ، ماضی نقلی ، ماضی بعید ، ماضی التزامی ، مضارع اخباری} **Na+{past tense, past perfect continuous, Present perfect tense}**	Other negative verbs

2.3.2 Stemming based on the similarity measure

A Persian stemmer that uses a dictionary and a list of prefixes and suffixes has been presented by Dianati et al. [22]. They used word structural similarity to stem the words. It is not necessary to have knowledge about suffixes and prefixes. The steps for the stemming processes are shown in Fig. 7.

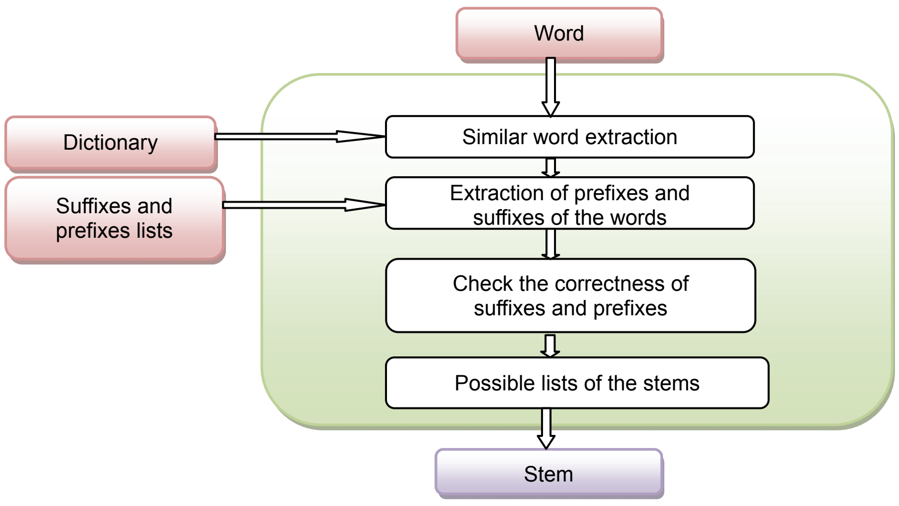

Word
Dictionary
Suffixes and prefixes lists
Similar word extraction
Extraction of prefixes and suffixes of the words
Check the correctness of suffixes and prefixes
Possible lists of the stems
Stem

Fig. 7. Steps of stemming process.

In similar word extraction, similar words are extracted from the dictionary based on their structural similarity. In this method, the similarity between words is defined based on numbers and the order of letters in a word. For example, the word “کتابهایمان” is "ایمان," “کتاب,” and “تاب.”

Affix lists were used to extract the suffixes and prefixes. The longest affix of a word is removed and the remaining part is the stem.

Five hundred words were selected from the per-Tree-Bank corpus [23] with their stems to test their algorithm and they claimed that 72.8% of the words were correctly stemmed for the Persian language. They used 500 words from the sortedtest.txt dataset to test English words and 71.2% of the words were stemmed correctly. The advantage of this algorithm is its independence of the language and the fact that there is no need to have Persian language morphological knowledge [22].

3. Evaluation of Persian Stemmers

We are proposing some new metrics to evaluate and compare Persian stemmers in information retrieval systems. Table 3 shows the results.

3.1 Performance

The performance of the information retrieval system is measured by precision and recall. To evaluate a stemmer, these two items are measured twice in a retrieval system: once with a stemmer and the other time, without any stemmers. Precision is the number of retrieved documents that are relevant to the query [23]:

$$Precision = \frac{\#(Relevant\ documents\ retrieved)}{\#(Retrieved\ documents)} = P\,(relevant\ documents\ retrieved \mid retrieved\ documents) \tag{7}$$

And recall is the numbers of relevant documents that are retrieved [24, 25]:

$$Recall = \frac{\#(Relevant\ documents\ retrieved)}{\#(Relevant\ documents)} = P\,(relevant\ documents\ retrieved \mid relevant\ documents) \tag{8}$$

We found which Persian stemmer increased the precision or recall items of information retrieval systems and indicated this by making a mark in a Table 3.

3.2 Size of Test Collection

To test the information retrieval system, a test collection, which is a collection of documents and queries, is needed [16]. Some of the designers of Persian stemmers used famous test collections and some of them collected documents and queries and tested their stemmer by using them. We tried to find out the number of documents and queries to show if they tested their algorithm by using a large collection or not.

3.3 Dependence on the Language

Some stemmers were created only for Farsi and some of them do not depend on any language and can work for more than one language.

Table 3. Evaluation of Persian stemmers

	Size of test collection	Performance in IR system		Dependence to the language
		Precision	Recall	
Structural Approaches				
Farsi stemmer1	small	√	√	High
Farsi stemmer2	medium	√	√	High
Bottom up approach stemmer	medium	√		High
Structural rule-based stemmer	large	√		High
Perstem	large	√	√	High
Lookup Table				
Bon stemmer	small		√	High
Improved Krovetz algorithm	small	√		High
Stemming based on similarity measure	small			No
Statistical Approaches				
Statistical Farsi stemmer	small	√		Low

As shown in Table 3, some of the stemmers increase the precision of an information retrieval system, some increase the recall of the system, and some increase both. There are some stemmers that can do a good job of finding the stems of a word, but they have not been tested on an information retrieval system. As such, there is not any information about the effect of these stemmers on the performance of an information retrieval system. We should pay attention to the size of the datasets so that the stemmers can be compared to each other. This is important because although these stemmers increase the recall or precision of an information retrieval system, the size of the dataset was small and the results change if it is tested on a larger dataset.

4. Conclusion

In this paper we studied and analyzed various Persian stemmers. First, we carefully described each algorithm and presented the strengths and weaknesses of each one. We classified these algorithms into three common approaches: structural, lookup, and statistical approaches. As we described before, there are few Persian stemmers and most of them use the structural approach. We have proposed some new metrics and tried to compare and evaluate Persian stemmers based on them. The results showed that most of the stemmers used small test collections and that they need to be tested on large test collections.

References

[1] E. Rahimtoroghi, H. Faili, and A. Shakery, "A structural rule-based stemmer for Persian," in *Proceedings of 2010 5th International Symposium on Telecommunications (IST),* Tehran, Iran, 2010, pp. 574-578.

[2] A. A. Sharifloo and M. Shamsfard, "A bottom up approach to Persian stemming," in *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP),* Hyderabad, India, 2008, pp. 583-588.

[3] F. Momenipour and M. Keyvanpour, "Analytical study of various information retrieval models based on mathematical approaches," *Journal of Next Generation Information Technology*, vol. 4, no. 5, pp. 63-73, 2013.

[4] M. Tashakori, M. Meybodi, and F. Oroumchian, "Bon: the Persian stemmer," in *EurAsia-ICT 2002: Information and Communication Technology.* Heidelberg, Germany: Springer, 2002, pp. 487-494.

[5] M. Agosti, M. Bacchin, N. Ferro, and M. Melucci, "Improving the automatic retrieval of text documents," in *Advances in Cross-Language Information Retrieval.* Heidelberg, Germany: Springer, 2003, pp. 279-290.

[6] R. Karimpour, A. Ghorbani, A. Pishdad, M. Mohtarami, A. AleAhmad, H. Amiri, and F. Oroumchian, "Improving Persian information retrieval systems using stemming and part of speech tagging," in *Evaluating Systems for Multilingual and Multimodal Information Access.* Heidelberg, Germany: Springer, 2009, pp. 89-96.

[7] P. Janarthanan and N. Rajkumar, "Information Retrieval Using Second Order Co-occurrence PMI," *International Journal of Information Technology & Computer Science*, vol. 9, no. 3, pp. 1-10, 2013.

[8] S. Estahbanati and R. Javidan, "A new stemmer for Farsi language," in *Proceedings of 2011 CSI International Symposium on Computer Science and Software Engineering (CSSE),* Tehran, Iran, 2011, pp. 25-29.

[9] S. Estahbanati and J. Reza, "A new multi-phase algorithm for stemming in Farsi language based on morphology," *International Journal of Computer Theory and Engineering*, vol. 3, no. 5, pp. 623-627, 2011.

[10] K. Taghva, R. Beckley, and M. Sadeh, "A stemming algorithm for the Farsi language," in *Proceedings of 2005 International Conference on Information Technology: Coding and Computing (ITCC)*, Las Vegas, NV, 2005, pp. 158-162.

[11] A. Mokhtaripour, and S. Jahanpour, "Introduction to new Farsi stemmer," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, Arlington, VA, 2006, pp. 826-827.

[12] A. H. Jadidinejad, F. Mahmoudi, and J. Dehdari, "Evaluation of PerStem: a simple and efficient stemming algorithm for Persian," in *Multilingual Information Access Evaluation I: Text Retrieval Experiments.* Heidelberg, Germany: Springer, 2010, pp. 98-101.

[13] J. Dehdari and D. Lonsdale, "A link grammar parser for Persian," in *Aspects of Iranian Linguistics*. Newcastle upon Tyne: Cambridge Scholars Publishing, 2008, pp. 19-33.

[14] A. AleAhmad, H. Amiri, E. Darrudi, M. Rahgozar, and F. Oroumchian, "Hamshahri: a standard Persian text collection," *Knowledge-Based Systems*, vol. 22, no. 5, pp. 382-387, 2009.

[15] J. Mehrad and S. R. Berenjian, "Providing a Persian language singular-stemmer system (RICeST Stemmer)," *International Journal of Information Science and Management*, vol. 9, no. 2, pp. 13-22, 2011.

[16] M. Melucci and N. Orio, "A novel method for stemmer generation based on hidden Markov models," in *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM)*, New Orleans, LA, 2003, pp. 131-138.

[17] M. M. Nasiri, K. S. Esmaeili, and H. Abolhassani, "A statistical stemmer for Persian language," in *Proceedings of 11th International CSI Computer Conference (CSICC2006)*, Tehran, Iran, 2006.

[18] M. Bacchin, N. Ferro, and M. Melucci, "A probabilistic model for stemmer generation," *Information Processing & Management*, vol. 41, no. 1, pp. 121-137, 2005.

[19] Iranian Students' News Agency, http://www.isna.ir/en.

[20] Apache Lucene, http://lucene.apache.org/core/index.html.

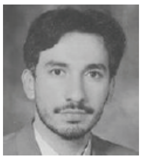
[21] R. Hesamifard and G. Ghassem-Sani, "A stemming algorithm for the Persian words," in *Proceedings of the 11th Annual International CSI Computer Conference (CSICC2006)*, Tehran, Iran, 2006, pp. 515-519.

[22] M. H. Dianati, M. H. Sadreddini, A. H. Rasekh, S. M. Fakhrahmad, and H. Taghi-Zadeh, "Words stemming based on structural and semantic similarity," *Computer Engineering and Applications Journal*, vol. 3, No. 2, pp. 89-99, 2014.

[23] M. Ghayoomi, "Bootstrapping the development of an HPSG-based Treebank for Persian," *Linguistic Issues in Language Technology*, vol. 7, no. 1, pp. 1-13, 2012.
[24] M. Keyvanpour and R. Tavoli, "Document image retrieval: algorithms, analysis and promising directions," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 1, pp. 93-106, 2013.
[25] M. Keyvanpour and R. Tavoli, "Feature weighting for improving document image retrieval system performance," *International Journal of Computer Science Issues*, vol. 9, no. 3, pp. 125-130, 2012.

Fatemeh Momenipour Moghadam

She received her B.S. in software engineering from Islamic Azad University, Guilan, Iran. She received her M.s in software engineering at Islamic Azad University, Qazvin Branch, Qazvin, Iran. Her research interests include information retrieval and Persian stemming.



MohammadReza Keyvanpour

He is an Associate Professor at Alzahra University, Tehran, Iran. He received his B.S. in software engineering from Iran University of Science & Technology, Tehran, Iran. He received his M.S. and Ph.D. in software engineering from Tarbiat Modares University, Tehran, Iran. His research interests include information retrieval and data mining.