A New Multilingual Stemmer to Improve the Effectiveness of Text Categorization and Information Retrieval

[1] Said Gadri, [2] Erich Neuhold
[1]Department of Computer Science, University Mohamed Boudiaf of M'sila, Algeria
[2]Department of Computer Science, University of Vienna, Austria

kadri.said28@gmail.com; erich.neuhold@univie.ac.at

ABSTRACT

Information retrieval IR is the process of finding information (generally documents) that matches the needs of the user. One way to improve the search effectiveness as well as the quality of text categorization is to build an effective stemmer that helps to match user's queries with relevant documents in IR and reduce the space of textual representation in TC. This has been always an interesting research topic in IR and TC. We can define stemming as the process of reducing inflected and derived words to their reduced forms (stems or roots). Many stemmers have been developed for different languages, but there is always many weakness and problems. In the present work, we have developed a multilingual stemming approach, based on the extraction of the word root and that exploits the technique of n-grams of characters. Our experiments have been done on three languages which are: Arabic, English and French.

Keywords: Information retrieval, Machine learning, Natural language processing, Root extraction, Stemming.

INTRODUCTION

Text categorization TC is an interesting task in text mining field. It is useful in several applications such as: spam filtering, assignation of web pages to yahoo groups, etc. it becomes more and more important as the amount of texts in electronic format grows every day. This task is done by assigning a set of texts to another set of predefined categories (classes). To fulfil TC requirements, there exist two approaches: manual approach (human experts) which is focused around manual development of classification rules, machine learning approach which is based on the use of algorithms known in machine learning field such as: K-NN, NB, Decision trees, NB, SVM, ANN, , etc. During TC process, the document must pass through many steps: pre-processing step which consists of removing irrelevant words such as punctuation and stop words, representation step which consists of representing each document with a vector of terms (words, phrases, n-grams, etc), and finally the calculation step in which we calculate term frequencies TF, and inverse document frequencies TF-IDF. Unfortunately, one critical problem can be encountered during the representation step which is the large size of vectors used to represent these documents, this case rises when we use big corpora of texts. To overcome this problem, a set of statistical methods can be used to select the most relevant terms and use them in the entry of learning algorithms (Mesleh, 2007; Ionan, 2008; Komkid, Narodom, Kittisak and Nettaya, 2012). Another technique that gives good results for the categorization of Arabic texts is the selection of relevant terms using stemming technique.

In the field of Information Retrieval, the stemming is also used to conflate a word to its different forms in order to avoid inconsistency between the words occurring in the documents and the query asked by the user. For example, if the user is looking for a document on "How to draw" and he submits a query on "drawing", he may not obtain all results he is looking for. But, if his query is stemmed, so that "drawing" becomes "draw", then

retrieval will be more successful. Thus, if we want to give a simple definition of stemming, we can say that stemming is a linguistic technique that permits to replace a large number of terms (words) occurring in a collection of documents and semantically close by their reduced forms (roots or stems), the main objective is to reduce the size of the vector of terms on one hand, and to improve the accuracy of categorization in TC and the effectiveness of search in IR on the other hand.

Several stemmers have been built for different languages such as: English, French, Spanish, Italian and Arabic. Each one has its own weaknesses as well as its disadvantages. We note also, that most of these stemmers are language dependent (Hadni, Ouatik and Lachkar, 2013). So, it seems useful to design and develop a new stemming algorithm which is totally language independent. For this purpose, we developed in our study a multilingual stemming approach, that relies on the search of the word root. This approach has been validated on three different languages namely: English, Arabic and French and gave promising results.

We also note that our proposed algorithm can work well for the lemmatization process, but we have focused here only on the stemming process which is our first concern. The present paper is organized as follows: section 2 presents some concepts related to the stemming process. Section 3 is a detailed overview of the related works. In the fourth section we give some metrics which permit to compare between stemmers. In section 5, we describe our proposed stemming approach. The sixth section illustrates the experimental part we have done to validate our approach, as well as the obtained results. In section 7, we discuss the results obtained in the previous section. Section 8 displays a comparison established between our proposed multilingual stemmer and the most popular ones in the field. In the last section, we summarized the realized work and suggested some perspectives for future researches.

BASIC CONCEPTS

Stemming and Lemmatization

Stemming and lemmatization have almost the same function. Both of them replace a word by its basic form; a 'root' or 'stem' for stemming and 'lemma' for lemmatization. Therefore, there exist a main difference between both concepts. In stemming the (root/stem) is obtained after applying a set of rules (removal of affixes) without requiring a grammatical analysis on the text. In contrary, the lemmatization process is consisting in obtaining the 'lemma' of a word which requires replacing conjugated verbs by their infinitive forms and nouns by their singular forms after applying a grammatical analysis on the text (e.g., POS tagging). So, it is a more complicated operation than stemming especially when implementing it on machine.

Over Stemming and Under Stemming Errors

In stemming, two errors may be encountered: over stemming and under stemming. We talk about over-stemming if two words having different stems are stemmed to the same root/stem. similarly, we talk about under-stemming if two words that should be stemmed to one root/stem are not. Both errors influence negatively on the performance of the stemmer. We note also that, when a word is under-stemmed, it becomes enough difficult to determine if two different words are related according to their morphology or not. In the case of over stemming, the major problem is that it will be possible that two no related words which but share a common part of their morphological root are incorrectly detected as related because their stems are identical.
Many researchers have proved that light-stemming decreases the over-stemming errors but increases the under-stemming errors. In contrast, heavy stemming reduces the under-stemming errors but increases the over-stemming errors (Chris, 1990, 1994).

Conflation Methods

The terms conflation is generally used to denote the fact of matching morphological term variants. We distinguish two kinds of conflation; manual conflation based on some regular expressions and automatic conflation based essentially on some programs called stemmers. Figure 1 explains briefly the different methods of conflation.

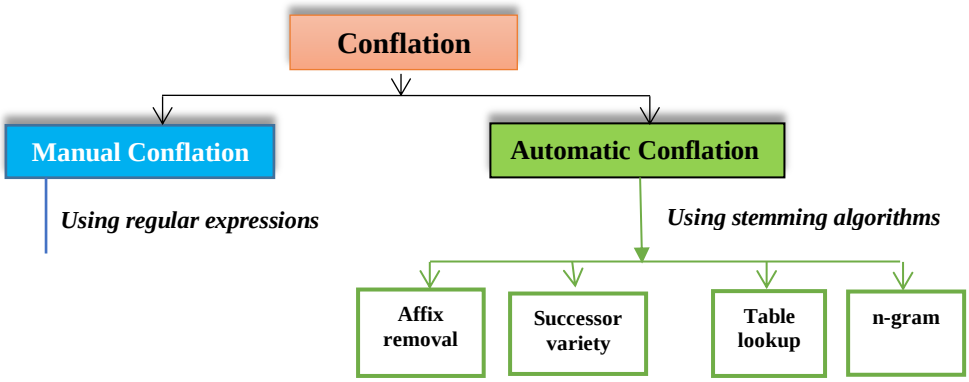

Figure 1. Conflation methods

Affix removal methods

This class of methods removes affixes (suffixes, infixes, prefixes) from the words subject of conversion into a common form called stem. Most stemmers are based on the affix removal methods. They are also based on two main principles: the first is iterations and the second is the longest match possible. We define an iterative stemming algorithm as a recursive process that removes strings in each order-class, starting from the end of the word toward its beginning. This approach of stemming does not allow more than one match within a single order-class (Deepika, 2012). The meaning of longest-match is that within any given class of endings, if more than one ending provides a match, the longest one should be removed.

Successor variety method

This class of methods generally uses the frequencies of letter sequences in a given body text as the basis of stemming algorithm. In simple terms, the successor variety of a string can be defined as the number of different characters that follow it in words in some body of text (Hafer & Weiss, 1974). For example, we take the following body of text consisting of many words: (call, cake, cease, chair, cash, cost, cheap, core, check, cubic). To find the successor varieties for the word "carpenter" for example, we must use the following process. The first letter of “carpenter” is "c". "c" is followed by five characters in the text body: "a," "e,”, “h,”, “o,” and "u." Thus, the successor variety of "c" is five (05). The next successor variety for “carpenter” would be three (03), since “ca” is followed by “l,”, “k,”, “s." in the text. if this process is performed using a large body of text, the successor variety of substrings of a term will decrease as more characters are added until a segment limit is obtained. At this level, the successor variety will consequently increase. This kind of information will be used to identify stems. We can also apply the same process for Arabic language. For example, if we have the following body of text: (علم، عبق، عائق، علو، عمق، علوم، عوام، علوج، علومه). The successor varieties of the word “علومهم” will be as follows: "(01) "علوم," ,(02) "علو," ,(02) "عل," ,(05) "ع,).

Table lookup method

Some stemming algorithms use a table of index terms and their stems stored in advance to extract the stem of a word that occurs on the index. Therefore, terms from queries and indexes could then be

stemmed based on this table lookup (Frakes, 1984) using a well-known technique such as: B-tree or Hash table, such lookups would be very fast. For example, “used”, “users”, “using”, “useful” all these terms can be stemmed to the common stem “use”. with this approach, two problems can be met. The first is that to build these lookup tables we need to work directly on a language and thus requires prior linguistic knowledge. The second, is that these tables may miss out some exceptional cases. A third problem is the storage cost for such a table (the required memory space).

Table 1.

Principle of Table Lookup Method

Term	Stem
English	
Information	Inform
Informatics	Inform
Inform	Inform
Informational	Inform
Arabic	
علوم	علم
عالم	علم
معلم	علم
علماء	علم
معلمة	علم
معلمات	علم
علماء	علم
يتعلّمون	علم

N-Grams method

Proposed by Adamson and Bareham (1974) and also called the shared digram method. A digram (a bigram) is a pair of consecutive letters, but we can also use 3-grams, 4-grams, …, and hence it’s called n-grams method. With this method, two words are associated on the basis of common unique digrams they both share. To calculate this association measure, we generally use Dice’s coefficient (Frakes, 1992). For example, the terms; policy and political can be segmented into bigrams as follows:

policy => po ol li ic cy (05 digrams)

unique digrams = po ol li ic cy (05 digrams)

political => po ol li it ti ic ca al (08 digrams)

unique digrams => po ol li it ti ic ca al (08 digrams)

common unique digrams => po ol li ic (04 digrams)

Thus, " policy " has (05) digrams, all are unique, and " political " has (08) digrams, all are unique. The two words share (04) unique bigrams.

Once the common unique bigrams for the word pair have been identified and counted, a similarity metric based on them is computed. The first used similarity metric was Dice's coefficient, which is defined as follows: (Frakes, 1992)

$$S = 2 * C/(A + B). \quad (1)$$

where A is the number of unique bigrams in the first word, B the number of unique bigrams in the second, and C the number of unique bigrams shared by A and B.

For the example above, Dice's coefficient would equal: (2 x 4)/(5 + 8) = 8/13 = 0.61. Such similarity metrics are determined for all pairs of terms in the database. Once such similarity metric is computed for all the word pairs they are clustered as groups.

LITERATURE REVIEW: STEMMING ALGORITHMS

Stemming algorithms can be classified in three main groups: truncating methods, statistical methods, and mixed methods (hybrid methods) (Jivani, 2011). Each group has its own way to find the stem of the word.

Truncating Methods

This group of methods is related to removing the affixes of a word. One well-known stemmer in this group is the Lovins stemmer proposed by Lovins in 1968. It performs a lookup on a table that contains a number of endings, conditions and transformation rules (294 endings, 29 conditions and 35 transformation rules) (Lovins, 1968). The Lovins stemmer always proceeds to the removal of the longest suffix from a word. And hence, the word is recoded using a separate table that makes various adjustments to convert these stems into correct words. As advantages of this algorithm, we note that it is very fast and can handle the removal of double letters in words like ‘setting’ being transformed to ‘set’ and also handles many irregular plurals like – appendix and appendices, etc. The drawbacks of the Lovins approach are that it is time and data consuming. Furthermore, it is possible that many suffixes are not available in the table of endings.

Porter stemming algorithm (Porter 1980, 2001) is one of the best and most popular stemming methods until our days, proposed in 1980 by porter. Many improvements have been introduced on the basic algorithm. The main idea of this stemmer is that the suffixes in English are often made up of a combination of smaller and simpler suffixes. It works through five steps, within each step, many rules are applied until one of them verify the required conditions. If the selected rule is accepted, the suffix is removed accordingly, and the stemming process pass to the next step until achieving the fifth step in which the resultant stem is returned. Porter have also built a detailed framework of stemming which is known as ‘Snowball stemming framework’. This framework allows programmers to develop their own stemmers for different languages. The Paice/Husk stemmer is an iterative algorithm based on a table containing a large list of rules (about 120 rules) indexed by the last letter of a suffix (Chris, 1994). On each iteration, it tries to find a valid rule according to the last character of the word. Each rule performs one of two possible tasks, a deletion or replacement of an ending. If there is no such rule, the algorithm terminates. There is another case for which the algorithm also terminates, it is when a word begins with a vowel and there exist only two letters left or if a word begins with a consonant and there exist only three characters left. Otherwise, the rule is applied and the process repeats. As advantages we note: it is a simple stemmer, and every iteration performs both deletion and replacement according to the applied rule. The disadvantage is it is a very heavy algorithm and over stemming may occur in some cases. Dawson Stemmer (Dawson, 1974): This stemmer is an extension of the Lovins algorithm except that it covers a much more comprehensive list of suffixes (about 1200 suffixes). Like Lovins, it is a single pass stemmer and

hence it is relatively fast. The suffixes are indexed by their length and last letter and then stored according to reversed order. Therefore, they are organized as a set of character trees which permit a rapid access. The advantage is that it covers more suffixes than Lovins stemmer and it is fast in execution. The disadvantage is it is very complex and does not have a unique standard implementation.

Statistical Methods

This group of methods is called statistical methods, it contains stemmers which are based on statistical analysis and techniques. Most of the methods remove also the affixes but after implementing some statistical procedures. In this group we can find the following stemmers:

N-Grams Stemmer (Chris, 1994; Croft & Xu, 1998): it is considered as a language independent stemmer in which a string-similarity approach is used to convert word inflation to its stem. An n-gram can be defined as a set of n successive characters extracted from a word. The basic idea of this approach is that, similar words share a high proportion of n-grams. For n equals to 2, the words extracted are called digrams. For example, the word 'information' results in the generation of the digrams: *i, in, nf, fo, or, rm, ma, at, ti, io, on, n* Where '*' denotes a padding space. usually, n can take a value of 4 or 5. The major advantage of this stemmer is that it is a completely language independent and hence very useful in a large range of applications. The disadvantage is it is memory space consuming when creating and storing these n-grams. HMM Stemmer: This stemmer was proposed by Melucci and Orio (2003) and mainly based on the concept of the Hidden Markov Model (HMMs) which is a finite-state automata. At each transition, the new state provides a character with a calculated probability. This method does not need a prior linguistic knowledge of the dataset, and the probability to take each path can be computed and the most probable path is found using the Viterbi coding in the associated automata graph. YASS stemmer (Yet Another Stripping Stemmer) was proposed by Prasenjit and Majumder et al (Majumder et al., 2007). It is considered as a statistical and corpus-based stemmer in the same time, since it does not rely prior linguistic knowledge. It seems effective for languages that are suffixing in nature, and its performance is comparable to of some well-known stemmers such as Porter and Lovins in terms of average precision and the total number of relevant retrieved documents. In YASS approach, a set of boolean string distance measure is defined (Levenstein, 1966), this distance measure is used to check similarity between two words A and B by mapping them to a real number R, where a smaller value of R indicates greater similarity between A and B.

Mixed Methods

This group contains a variety of mixed methods which are:

The Inflectional and Derivational Methods: they involve both the inflectional and the derivational morphology analysis, require a very large corpus to develop these types of stemmers and hence they are part of corpus base stemmers. In inflectional methods, the word variants are mainly related to the variation of language syntax like plural, gender, case, etc. Whereas, in derivational methods, the word variants are related only to the part-of-speech (POS) of a sentence in which the word occurs (Jivani, 2011).

Krovetz Stemmer (KSTEM): was presented in 1993 by Robert Krovetz (1993), it is a linguistic

lexical stemmer. It is very complicated in nature, since it is based on the inflectional property of words and the language syntax. It effectively removes inflectional suffixes in three steps. Unfortunately, Krovetz stemmer does not find the stems for all word variants, that is why it is generally used as a pre-stemmer before applying any other stemming algorithm namely: Porter, Lovins, which can increase the speed and the effectiveness of the main stemmer. Krovetz stemmer is considered as a light stemmer comparing it with other stemmers and does not produce a good recall and precision performance.

Xerox Inflectional and Derivational Analyzer: the linguistics groups at Xerox have developed a lexical database for English and some other languages which can analyze and generate inflectional and derivational morphology. The inflectional database reduces each surface word to the form which can be found in the dictionary, as follows (Hull & Grefenstette, 1996): nouns singular (e.g. children child), verbs infinitive (e.g. understood understand), etc. The advantages of this stemmer are that it works well with a large document and also removes the prefixes where ever applicable. The obtained stems are valid since a lexical database providing a morphological analysis of any word in the lexicon is available. Among the drawbacks of this stemmer is that the output depends mainly on the lexical database which may not be exhaustive. Another drawback, is that this method is based on a lexicon, hence, it cannot always give the correct stems which does not appear in the lexicon. We also note, that this stemmer has not been implemented successfully for many languages.

Corpus Based Stemmer This method of stemming was proposed by Xu and Croft (1998). This method tries to overcome some of the drawbacks of Porter stemmer. The corpus-based stemming refers to automatic modification of conflation classes – words that have resulted in a common stem, to suit the characteristics of a given text corpus using statistical methods. The main hypothesis is that word forms that should be conflated for a predefined corpus will co-occur in documents of the same corpus. Using this concept, many errors of over stemming and under stemming can be resolved e.g. It is also important to note that this stemmer works within two steps, in the first step, it uses the Porter stemmer to identify the stems of conflated words and in the second step, it uses the corpus statistics to redefine the conflation. The advantage of this method is it can potentially avoid making conflations that are not appropriate for a given corpus and the result is an actual word not an incomplete stem. The disadvantage is that you need to develop the statistical measure for every corpus separately and the processing time increases as in the first step, and stemming algorithms are first used before using this method.

The Context Sensitive Stemmer is done using statistical modelling on the query side. This method was proposed by Funchun Peng et al (2007). The basic idea behind this stemmer is that before submitting any query to the search engine, we predict morphological variants which would be useful in the search process. Using this method, the number of bad expansions can be reduced, which in turn reduces the cost of additional computation and consequently improves the precision at the same time. After the derivation of the predicted word variants from the query, a context sensitive document matching is done for these variants. The major advantage of this stemmer is that it improves selective word expansion on the query side and matches conservative word occurrence on the document side. Two disadvantages can arise here, the processing time and the complexity of the algorithm.

Many Stemming algorithms have been developed for a wide range of languages including English (Greengrass et al., 1996), Latin (Hull, 1996), Swedish (Carlberger et al., 2001), German and Italian (Monz & Rijke, 2001), French (Moulinier et al., 2001), Turkish (Ekmekcioglu et al., 1996; Tan & Yu, 2000).

For Indian languages (Thangarasu & Manavalan, 2013; diBijal & Sanket, 2014; Kasthuri & Kumar, 2014), we can note: Punjabi (diBijal, 2014) with an accuracy of 80.73%, Bengali (Suprabhat & Pabitra, 2011) gives an accuracy of 96.27%, Urdu (Kasthuri, 2014) based on two approaches: Length based and Frequency based, gives for each approach successively one of the following accuracy: 84.27% and 79.63%, Hindi (Thangarasu, 2013) gives an accuracy of 91.59%, Another Hindi stemmer was proposed in 2014 by Vishal Gupta (2014), based on Suffix Stripping of porter and gives an accuracy of 83.65%. For other non-Indian languages, we can note: Persian, Indonesian (Husain, 2012; diBijal, 2014; Kasthuri, 2014).

For Arabic Language, there are three different Stemming approaches: the root-based approach (Al-Fedaghi & Al-Anzi, 1989; Khodja, 1999; Larkey & Connell, 2002; Momani & Faraj, 2009; Al-Nashashibi et al., 2010; AbuHawas & Keith, 2014); the light stemmer approach (Al-shalabi et al., 2003; Kanaan et al., 2005; Al-Nashashibi et al., 2010; Mohamad et al., 2010; Al-omari, et al., 2013); and the statistical stemmer approach (N-Grams) (Mustafa & Al-Radaideh, 2004; Hmeidi et al., 2010; Yousef et al., 2014). But, until now, no a perfect stemmer for this language is available. Figure 2 summarizes the different stemming algorithms.

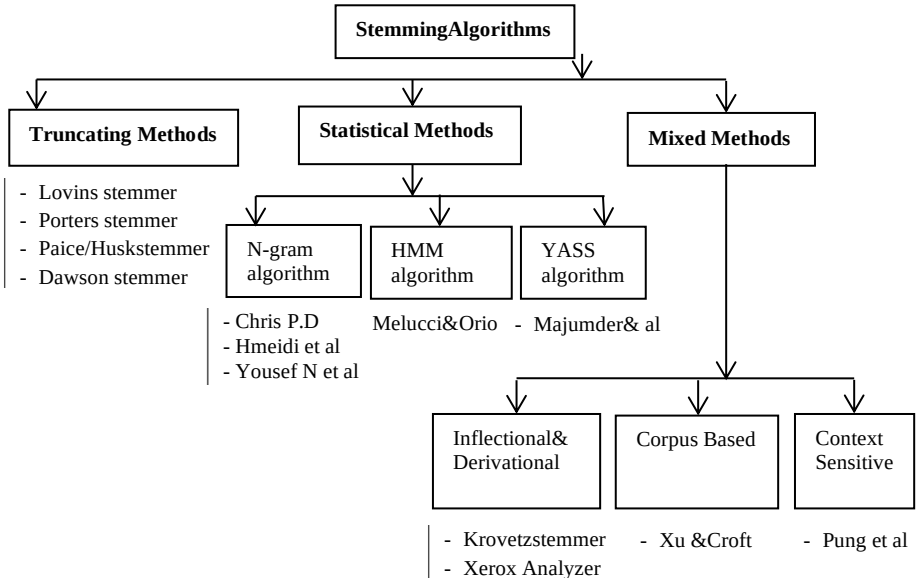

Figure 2. Classification of stemming Algorithms (Jivani, 2011)

COMPARISON AND EVALUATION OF STEMMERS

Moral, Antoni, Imbert and Ramirez (2014) present a detailed survey of stemming algorithms in information retrieval covering: principle of stemming, stemming errors, stemming evaluation metrics, corpora used by researchers in IR. In this section we concentrate on most-known metrics used to compare between stemmers, notably:

Metric1: The Strength of a Stemmer

Generally, represents the extent to which a stemming algorithm reduces words to their stems (Frakes, 2003; Deepika, 2012). This measure depends on the rate of over-stemming and under-stemming made by a stemmer. According to this measure, there exist two kinds of stemmers, a strong (or heavy) stemmer which has a high rate of over-stemming errors, and a weak (or light) stemmer characterized by having a high under-stemming rate (Moral et al., 2014). Some researchers define the stemmer strength as the average number of words per conflation class. Formally, it can be expressed as follows: (Paik et al., 2011)

$$Na = Nw/Ns. \quad (2)$$

Where:
N_a: the average number of words per conflation class.
N_w: the number of distinct words before stemming.
N_s: the number of unique stems after stemming.

In the same way, Paice (1994) proposed some parameters to calculate the strength of a stemmer: the under-stemming index (UI), the over-stemming index (OI), the stemming weight (SW) and an error rate relative to truncation (ERR). Based on these parameters, the strength of a stemmer can be expressed using its over-stemming and under-stemming indexes: as follows

$$SW = OI/UI. \quad (3)$$

An experiment has been done on Lovins, Porter and Paice/Husk stemmers and proved that Paice/Husk has the highest rate of over-stemming and Porter the lowest. on the other hand, Porter stemmer makes more under-stemming errors than the others, with Paice/Husk being the one generating least errors of this type. As conclusion of this experiment, the Paice/Husk stemmer is the strongest stemmer, followed by Lovins which is considered a strong stemmer, and finally Porter, which is the weakest among the three.

Metric 2: Recall and Precision

The previous metric" strength" is not used to evaluate the accuracy of a stemmer, but it is used only to classify it according to its errors. Thus, to evaluate the accuracy of the stemming process two other metrics have been proposed by Berry, Kent, Luehrs and Perry (1955). These metrics are widely known in IR. The first metric is the recall which is the rate of relevant documents returned by the IRS among all relevant documents as an answer to a submitted query. The second one is the precision which is the rate of the relevant documents among all documents returned by the IRS as an answer to a query. Some authors have linked the performance of a stemming algorithm in terms of recall and precision to the strength of the stemmer, and thus to under-stemming and over-stemming (Harman, 1991; Chris, 1994; Frakes, 2003). They concluded that strong stemmers will generally, increase the recall, but they also decrease the precision. Accordingly, they also proved that weak stemmers are better in matching related

words, and then increasing the precision, but are more likely to avoid conflating related words because of under-stemming, thus decreasing the recall

Metric 3: Index Compression Factor (ICF)

Another metric is also used to evaluate a stemming algorithm based on its conflation rate, this metric is known as the Index Compression Factor (ICF), it gives us how much the stemmer is able to compress the vocabulary in input of the algorithm and then how much it decreases efficiently the storage capacity needed and consequently increases the efficiency of the information retrieval system which has to deal with a reduced dictionary. Many researchers in the field have proved that the compression of the vocabulary mainly depends on the strength of the stemmer too. Lennon, Pierce, Tarry and Willett (1988); Harman (1991); Chris (1994); Frakes And Fox (2003) have also proved that strong stemmers give a better ICF than weaker stemmers, because, they match more words.

Metric 4: Computation Time

It is another parameter that is used by researchers in the field of stemming to compare the performance of stemmers, the main idea is to compute the time from submitting a query to its processing and then having the final results.

Metric 5: Similarity Measure

Frakes (2003) proposed a similarity measure which allows to find the similarity between two different stemmers by comparing their returned results. This similarity is calculated using the inverse modified Hamming distance. The experiments done by researchers on many stemmers ("S" stemmer, Paice/Husk, Lovins, Porter,) show that Lovins and Paice stemmers are the most similar, while the Paice and "S" stemmers are the most different. A deep analysis on the similarity metrics of all the possible combinations of two stemmers, lets the researchers to prove that the similarity between stemmers is related directly to their strength. The summary of all the features presented above for the most known stemming approaches reinforces the idea that all the cited metrics depend mainly on the strength of different stemmers and consequently on the rate of under-stemming and over-stemming errors as it is presented in figure 3 below:

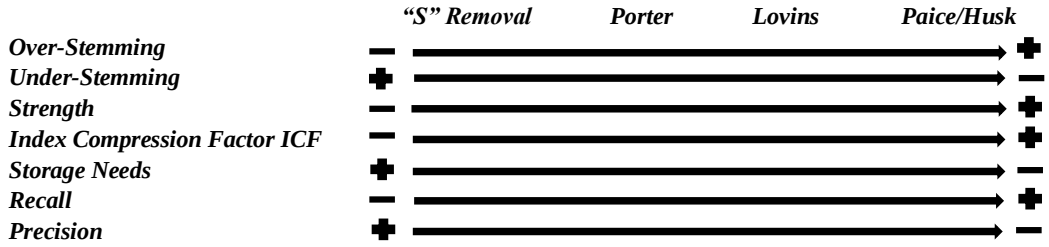

Figure 3. Features summary of classical stemmers

OUR STEMMING APPROACH

Our stemming approach is characterized by: the use of n-grams of characters technique and the extraction of the word root (Gadri & Moussaoui, 2015). In the experimental part, we have applied this approach on three different languages which are: Arabic, English and French. The stemming process can be divided into many phases:

Phase 1: segmentation of the word subject of stemming, and all the candidate roots into bigrams (2-grams). Table 2 describes the segmentation phase for three different words given in the three languages presented above.

Table 2.

Description of the segmentation phase (Phase 1)

Language	Word (W)/Bigrams	List of roots (Ri)/Bigrams
Arabic	يفقدون يف، فق، قد، دو، ون	R_1 = "نجح" ➔ (نج، جح) R_2 = "خرج" ➔ (خر ، رج) R_3 = "فقد" ➔ (فق، قد) R_4 = "عقد" ➔ (عق ، قد)
French	Connaissance co on nn na ai is ss sa an nc ce	R_1 = "command" ➔ (co, om, mm, ma, an, nd) R_2 = "avanc" ➔ (av, va, an, nc) R_3 = "concev" ➔ (co, on, nc, ce, ev) R_4 = "commenc" ➔ (co, om, mm, me, en, nc) R_5 = "connaît" ➔ (co, on, nn, na, aî, it) R_6 = "conclu" ➔ (co, on, nc, cl, lu)
English	Transaction tr ra an ns sa ac ct ti io on	R_1 = "action" ➔ (ac, ct, ti, io, on) R_2 = "extra" ➔ (ex, xt, tr, ra) R_3 = "intra" ➔ (in, nt, tr, ra) R_4 = "dict" ➔ (di, ic, ct)

Phase 2: Computation of stemming parameters which are:

N_w : The size of the word W in number of bigrams.
N_{R_i} : The size of the root R_i in number of bigrams N_{wR_i} : The number of common bigrams between the word W and the root R_i
$N_{w\bar{R}_i}$: The number of bigrams belonging to the word W and do not belong to the root R_i ($N_{w\bar{R}_i} = N_w - N_{wR_i}$)
$N_{R_i\bar{W}}$: The number of bigrams belonging to the root R_i and do not belong to the word W($N_{R_i\bar{W}} = N_{R_i} - N_{wR_i}$).

For the previous example we have the results shown in Table 3 below:

Table 3.

Computation of stemming parameters (phase 2)

Word(W)	N_w	Associated roots R_i	N_{R_i}	N_{wR_i}	$N_{w\bar{R}_i}$	$N_{R_i\bar{W}}$
يفقدون	05	R_1 = "نجح", R_2 = "خرج"	2, 2	0, 0	5, 5	2, 2
		R_3 = "فقد", R_4 = "عقد"	2, 2	2, 1	3, 4	0, 1
connaissance	11	R_1 = "command, R_2 = "avanc"	6, 4	2, 0	9, 11	4, 4
		R_3 ="concev",R_4= "commenc"	5, 6	4, 2	7, 9	1, 4
		R_5 = "connaît", R_6 = "conclu"	6, 5	5, 3	6, 8	1, 2
transaction	10	R_1= "action, R_2 = "extra"	5, 4	5, 2	5, 8	0, 2
		R_3 = "intra", R_4 = "dict"	4, 3	2, 1	8, 9	2, 2

Phase 3: Selection of candidate roots which are those sharing at least one bigram with the word W ($N_{wR_i} \geq 1$) among the predefined list of roots. In our previous example, we have obtained the results stored in Table 4:
Table 4.

Selection of candidate roots (phase 3)

Word(W)	N_w	**Associated roots R_i**	N_{R_i}	N_{wR_i}	$N_{w\bar{R}_i}$	$N_{R_i\bar{W}}$
يفقدون	05	R_3 = "فقد", R_4 = "عقد"	2, 2	2, 1	3, 4	0, 1
connaissance	11	R_1 = "command, R_3 ="concev",	6, 5	2, 4	9, 7	4, 1
		R_4= "commenc" , R_5 = "connaît",	6, 6	2, 5	9, 6	4, 1
		R_6 = "conclu"	5	3	8	2
transaction	10	R_1= "action, R_2 = "extra"	5, 4	5, 2	5, 8	0, 2
		R_3 = "intra", R_4 = "dict"	4, 3	2, 1	8, 9	2, 2

Phase 4: Computation of the distance *D (W, R_i)* between the word *W* and each candidate root R_i according to the formula (4):

$$\boldsymbol{D(W, R_i) = \frac{N_{w\bar{R}_i} + N_{R_i\bar{W}}}{N_w + N_{R_i}}} \quad (4)$$

In the case of our previous example, we obtain the results shown in Table 5 as follows:

Table 5.

Computation of the distance ***D (W, Ri)***

Word(W)	N_w	**Associated roots R_i**	N_{R_i}	N_{wR_i}	$N_{w\bar{R}_i}$	$N_{R_i\bar{W}}$	**Dist.Val *D(W,R_i)***
يفقدون	05	R_3 = "فقد", R_4 = "عقد"	2, 2	2, 1	3, 4	0, 1	0.4285, 0.7142
connaissance	11	R_1 = "command, R_3 ="concev",	6, 5	2, 4	9, 7	4, 1	0,7647, 0,5
		R_4= "commenc", R_5 = "connaît",	6, 6	2, 5	9, 6	4, 1	0,7647, 0,4117
		R_6 = "conclu"	5	3	8	2	0,625
transaction	10	R_1= "action, R_2 = "extra"	5, 4	5, 2	5, 8	0, 2	0.3333, 0.7142
		R_3 = "intra", R_4 = "dict"	4, 3	2, 1	8, 9	2, 2	0.7142, 0.8461

Phase 5: assigning the root having the minimum value of distance *D (W, Ri)* to the word *W*.

In the previous example, the roots assigned to the given words are illustrated on Table 6:

Table 6.

Extraction of the word root (step 5)

Word *(W)*	**Extracted root*(R)***	**Effective root**
يفقدون	فقد	فقد
connaissance	connaît	connaît
transaction	action	action

EXPERIMENTAL RESULTS AND INTERPRETATION

In order to validate the new proposed stemming approach, we have performed many experiments on the three languages cited previously. These experiments were as follows:

Used Dataset

We have used for each language three corpora with different sizes: small corpus, middle corpus, and large corpus (see Table 7). In each there exist three files as described below:

1. The file of derived forms (gross words) which contains several morphological forms of words derived from a list of roots.

2. The file of roots containing a collection of roots, we note here that for Arabic language, these roots may be: 3-lateral, 4-lateral, 5-lateral, and 6-lateral (see Table 8). We underline also that some of them are vocalic roots which contain at least one vowel.

3.The file of golden roots containing the list of effective roots which may be matched to the words present in each corpus, this golden list has been established by an expert linguist and used as reference list, i.e., the calculation of the root extraction accuracy (success ratio) is done by comparison between the list of obtained roots (extracted by the system) and the reference list (established by the expert), . The extraction process and the obtained results are shown in Tables 9, 10 and 11 as well as figures 4, 5, 6, 7, 8 and 9.

Table 7.

The Used Dataset

Corpus	Language	Size of the derived words file	Size of the roots file	Size of the golden roots file
Small corpus	Arabic	50	25	50
	French	44	36	44
	English	92	56	92
Middle corpus	Arabic	300	150	300
	French	250	220	250
	English	450	180	450
Large Corpus	Arabic	2000	650	2000
	French	1500	600	1500
	English	1500	620	1500

Table 8.

Examples of Arabic roots (3-lateral, 4-lateral, 5-lateral, 6-lateral)

Trilateral roots	Quadrilateral roots	Quinquelateral roots	Hexalateral roots
زرع	أكرم	انطلق	استعمل
صنع	أعان	انكسر	استحسن
جمع	حطّم	اقتصد	اخشوشن
أبى	علّم	اجتمع	اعشوشب
سعل	طمأن	تنازل	اقشعرّ

Obtained Results

After applying our proposed stemming approach on the three languages: Arabic, French, English, we obtained the results presented in the tables and figures bellow:

Table 9.

Illustration of the obtained results after the segmentation phase

Word(W)	**N-grams**	**Nb.Ng (N_w)**	**Root**	**N-grams**	**Nb.Ng (N_{R_i})**
		Arabic			
يتعلّمون	يت تع عل لّ ّم مو ون	7	علّم	عل لّ ّم	3
اقتصاد	اق قت تص صا اد	5	اقتصد	اق قت تص صد	4
سنستدرجهم	سن نس ست تد در رج جه هم	8	درج	در رج	2
		French			
apprentissage	ap pp pr re en nt ti is ss sa ag ge	12	apprend	ap pp pr re en nd	6
calculatrice	ca al lc cu ul la at tr ri ic ce	11	calcul	ca al lc cu ul	5
connaissance	co on nn na ai is ss sa an nc ce	11	connaît	co on nn na aî ît	6
		English			
bicycle	bi ic cy yc cl le	6	cycle	Cy yc cl le	4
transactions	tr ra an ns sa ac ct ti io on ns	11	action	ac ct ti io on	5
geopolitics	Ge eo op po ol li it ti ic cs	10	geo	Ge eo	2

Table 10.

Extraction of word roots using the new stemming approach.

Word (W)	**Nearest Roots R_i**	**Nb.Common Bigrams (N_{wR_i})**	**Distance Values $D(W, R_i)$**	**Extracted Root**	**Correct Root**
		Arabic			
يتعلمون	كّلم ، علم ، علم ، علج	1 ، 1 ، 3 ، 2	0.77 , 0.77 , **0.4**, 0.6	علم	علم
سنستدرجهم	درج	2	**0.6**	درج	درج
		French			
Apprentissage	Apprend, assembl, assist, associ, assur, autoris, comprend	4, 1, 2, 1, 2, 1, 2	**0.55**, 0.9, 0.78, 0.89, 0.77, 0.9, 0.78	apprend	apprend
connaissance	avanc, command, commenc, concev, conclu, connaît	3, 2, 3, 4, 3, 4	0.64, 0.78, 0.68, **0.57**, 0.66, 0.57	concev	connaît
		English			
transactions	action, extra, intra, dict	5, 2, 2, 1	**0.28**, 0.69, 0.69, 0.83	action	action
geopolitics	Action, geo, poly	1, 2, 2	0.86, **0.66**, 0.69	geo	geo

Table 11.

Extraction of word roots: Global results.

Corpus	Language	Nb. Roots	Nb.Words	Correct Results	Wrong Results	Succes Rate (%)	Error Rate (%)
Small	Ar	25	50	47	03	94,00	06,00
Corpus	Fr	36	44	41	03	93,18	06,82
	En	56	92	85	07	92,39	07,61
Middle	Ar	150	300	235	65	78,33	21,67
Corpus	Fr	220	250	217	33	86,80	13,20
	En	180	450	411	39	91,33	08,67
Large	Ar	650	2000	1643	357	82,15	17,85
Corpus	Fr	600	1500	1173	327	78,20	21,80
	En	620	1500	1327	173	88,46	11,54

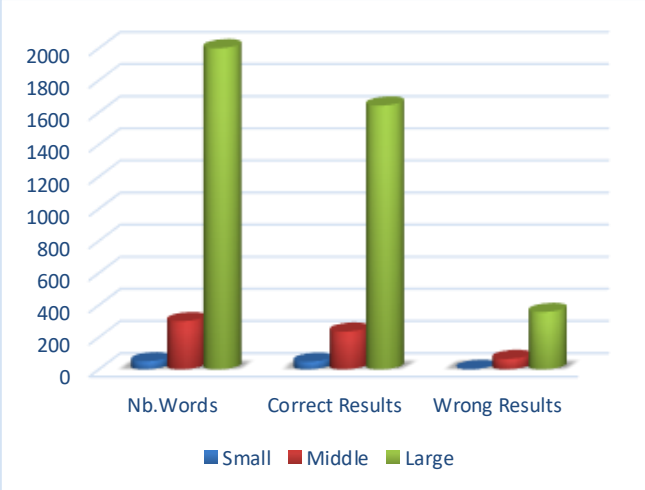

Figure 4. Illustration of the obtained results in number of words (Arabic)

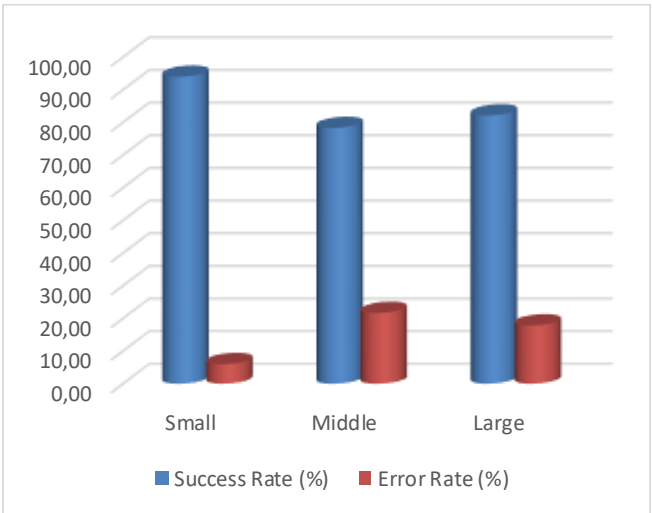

Figure 5. Computation of success and error rates (Arabic)

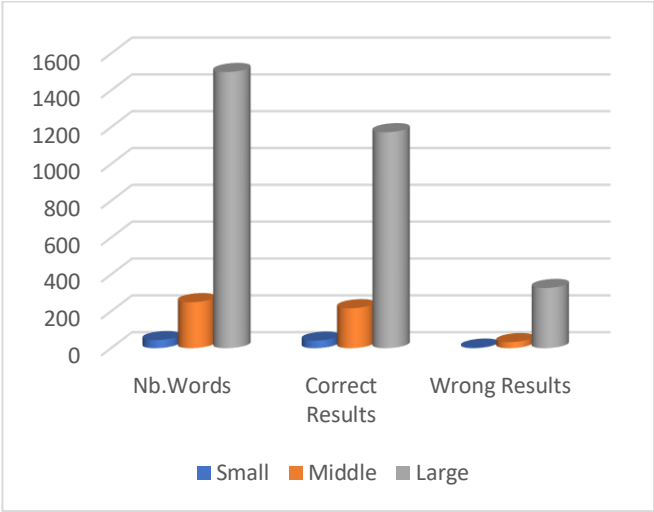

Figure 6. Illustration of the obtained results in number of words (French)

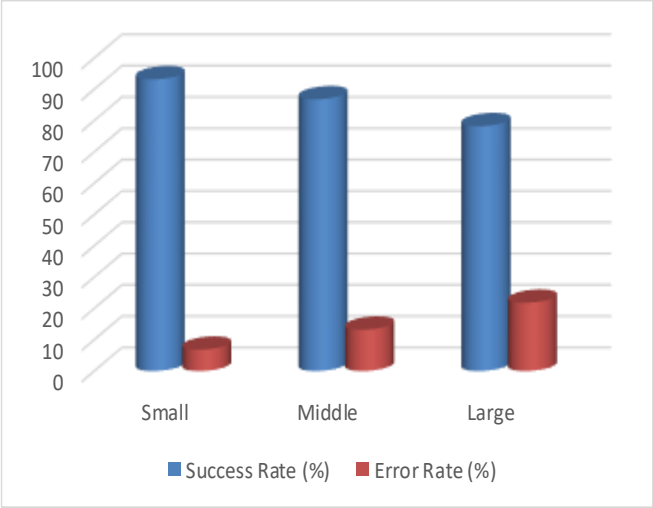

Figure 7. Computation of success and error rates (French)

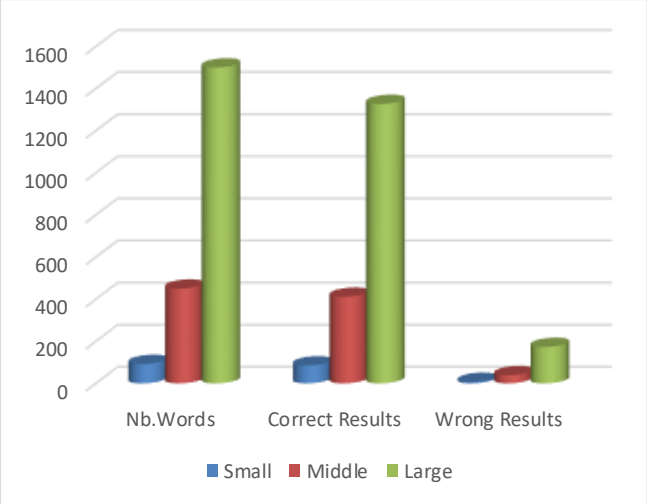

Figure 8. Illustration of the obtained results in number of words (English)

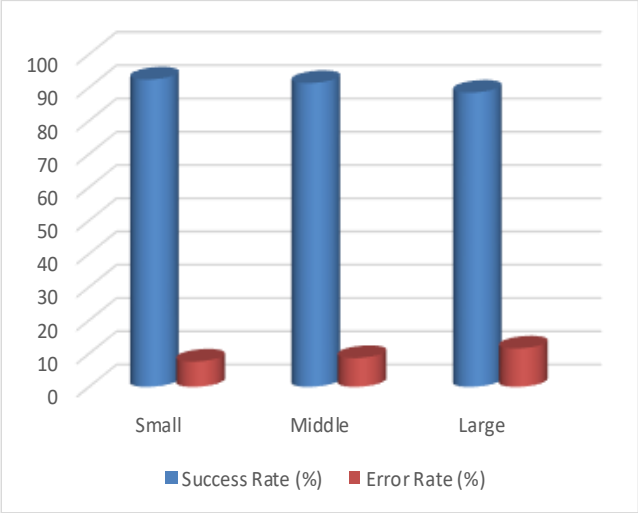

Figure 9. Computation of success and error rates (English)

Advantages of our Stemming Approach

Our new stemming approach has many advantages among them we can note the following:

1. It is language independent, so, we can apply it for many languages in the same time and without modification (a multilingual approach), Which is not the case for the existing stemmers, for example the Porter stemming framework "snowball" allows the researchers in different languages to develop their own stemmers (section 3), but they must modify the basic algorithm to deal with the morphology and grammatical rules of each language (a kind of language dependence).

2. Does not require any affix removal, where there is a real risk to remove some native letters of the word instead of the affixes.

3. Valid for any length of Arabic root. (e.g., 3-lateral, 4-lateral, 5-lateral, and 6-lateral roots).

4. Works very well for vocalic and strong roots, these two classes of roots usually pose problems in Arabic during their derivation, since they change their forms completely. We note here that no Arabic stemmer covers this property.

5. It is mainly based on a simple calculation of distances (it is a full statistical stemmer). Thus, no morphological rule or grammatical patterns are used during the extraction process.

6. It is a practical stemmer and then easy to implement.

7. Can be extended to lemmatization process.

DISCUSSION

In the first order, it seems important to underline that it was more interesting to establish a comparison between our multilingual stemming approach and some existing ones, but this was not possible, for a simple reason that all available stemmers are monolingual, i.e., each language has a set of stemmers based on a distinct theoretical background and different degrees of performances. This let's to conclude, that the major advantage of our stemming approach is that it is more general and completely language independent (Multilingual). On the other hand, we see on table 11, that the performance of stemming

algorithm in our approach decreases when the words base increases. This can be interpreted by the fact that the increasing of the words base must also be followed by the increasing of the roots base too. If no, many words will not be associated with their roots in the golden list (reference list of roots). A second remark that must be mentioned here, is the low value of accuracy related to Arabic stemming comparing with English and French, this is due to the method used to segment words and roots into bigrams of characters. For the method we have used in the present work, a bigram is obtained by concatenation of two direct successive letters. This will deal with Latin languages such as: English and French for which the root is a sequence of successive letters extracted from the beginning, the middle or the end of the word itself. E.g., for English we can meet pairs (word, root) as follows: (Geography, Geo), and the same for French (Confirmation, Confirm). This is completely different and will not deal with Arabic whose root must be built with a series of letters which cannot mandatory be consecutive, and consequently the number of common n-grams between the word and each candidate root decreases as well as the distance between them. It is the major reason for which it will be a large divergence between a word and its candidate roots.

as it is mentioned on the following examples: (كتاتيب, كتب) -(علماء, علم), where roots are formed of no consecutive letters. to overcome this problem, and so improve the stemming accuracy for Arabic, we suggest the following segmentation method: to form sequences of bigrams when segmenting a word or a root, we take all possible combinations of characters not only the pairs of successive ones: (e.g., علماء➔(عل، عم، عا، ع‌ء، لم،لا، ل‌ء، ما، م‌ء، اء).

COMPARISON WITH OTHER EXISTING STEMMERS

In this section, we establish a comparison of our proposed stemmer with some well-known stemmers as follows (Table 12):

Table 12.

Comparison with existing stemmers

Stemmer	**Type/Used approach**	**Advantages**	**Disadvantages**
Lovins stemmer	Truncating method (Affixes removal)	-Fast stemmer - Simple pass - Removal of double letters in words - Handles many irregular plural words	- Time consuming - Frequently fails to form words from stems - Depends on the used vocabulary
Porter stemmer	Truncating method (Affixes removal)	-Produces best results of stemming - Less error rates - It is a light stemmer	- Generated stems are not always the real words. -Works on several steps (05 steps). - Consumes more time.
Paice/Husk stemmer	Truncating method (Affixes removal)	- A simple stemmer - Each iteration considers both removal and replacement	- Heavy algorithm - Over stemming is possible
Dawson stemmer	Truncating method (Affixes removal)	-Covers a large list of suffixes (1200) -Fast in execution.	- enough complex - no standard implementation
Krovetz stemmer	Inflectional & derivational method	- It is a light stemmer. - Can be used as pre-stemmer	-Not valid for documents of large

		for other stemmers.	sizes -Enable to take care of words outside the lexicon -does not produce good results.
Youssef N et al stemmer (Arabic)	Statistical method (N-grams approach)	- Based on bigrams and string comparison - does not require morphological rules - Language independent	- Space consuming -Time consuming. -monolingual stemmer (Arabic).
Our proposed stemmer	Statistical method (An improved n-grams approach)	- Multilingual stemmer. -Based on an improved n-grams approach - does not require morphological rules - Language independent - For Arabic, it is valid for any length of root. (3, 4, 5, 6-lateral roots). - works well for strong and vocalic roots. - it is practical and then easy to implement. - It can be applied for lemmatization.	-Time consuming for big corpus. -Based on a reference base of roots.

CONCLUSION AND FUTURE SUGGESTIONS

In the present paper, we have studied one of the most critical problems in linguistics and NLP areas which is the problem of stemming and its direct influence on the quality of TC, IR, and many other fields of research such as: NLP and text mining tasks. This linguistic technique can reduce the size of terms vocabulary by 30 to 40% in TC, and increase the effectiveness of IRS. Throughout our article, we presented the well-known approaches, namely: morphological methods (truncating methods), statistical methods, and mixed methods that combine the two previous ones. For each class of methods, we have exposed briefly the main advantages and disadvantages. In the light of the presented methods, we proposed a multilingual stemming approach, which is characterized by: completely language independent, based on a statistical approach which in turn use the technique of n-grams, does not require any prior linguistic knowledge, having the capacity to find all kinds of roots (strong and vocalic), which is very perfect for Arabic since it is considered as one of the richest languages, with a complex morphology and a difficult structure. The realized experiments prove that the obtained accuracy when extracting the root for the studied languages is very satisfactory and can be improved in other future projects in the same field. Some of improvements that we can propose are as follows:

- Extend the present algorithm to lemmatization.
- Take in consideration other languages such as: German, Spanish, Turkish, etc.
- Apply our algorithm on other big datasets.
- Use another non-successive segmentation method for Arabic in order to accurate the extraction of the root as it was explained in discussion section.
- Improve the success rate for the three languages by introducing semantic processing.
- Use the developed stemmer to increase the quality of multilingual categorization in TC, and the effectiveness multilingual search in IR.

ACKNOWLEDGMENT

I thank in the first order the research team of the university of Vienna (Wien Universitat), Austria in which I have done a short internship during the period 1st, October – 10th, October 2018, especially Pr. Dr. WolfGang Klas the head of CS Department, Dr. Bilal AbuNaim, Ms. Elaheh M-O, Ms. Manuella the secretary of the department, For their good reception and very important advices during the period of my internship. I can't also forget all people who help me to achieve this work.

Non-funded

The present research project does not receive any particular grant from any public or private funding agency, neither in the commercial sector nor any other sectors.

REFERENCES

AbuHawas, F., & Keith, E. (2014). Rule- based Approach for Arabic Root Extraction: New Rules to Directly Extract Roots of Arabic Words. *Journal of Computing and Information Technology - CIT*, 22(1),68-57.

Adamson, G., Boreham, G. (1974). The Use of an Association Measure Based on Character Structure to Identify Semantically related pairs of words and document titles. *Information Storage and Retrieval*, 10(1), 253-60.

Al-Fedaghi, S., & Al-Anzi, F. (1989). A new algorithm to generate Arabic root-pattern forms. *In the 11th national Computer Conference and Exhibition* (pp. 400-391).

Al-Nashashibi, M.Y., Neagu, D., & Yaghi, A.A. (2010). An improved root Extraction technique for arabic words. In *2nd International Conference on Computer Technology and Development ICCTD* (pp.269-264). IEEE Xplore Press.

Al-Nashashibi, M.Y., Neagu, D., & Yaghi, A.A. (2010). Stemming techniques For Arabic words: A comparative study. In *2nd International Conference on Computer Technology and development CCTD* (pp.276-270).

Al-omari, A., Abuata, B., & Al-kabi, M. (2013). Building and Benchmarking New Heavy-Light Arabic Stemmer. In *4th International Conference on Information and Communication Systems ICICS* (pp.69-63).

Al-shalabi, R., Kanaan, G., & Al-Serhan, H. (2003). New Approach for Extracting Arabic Roots. In *the International Arab Conference on Information Technology ACIT03* (pp.59-42). Alexandria, Egypt.

Carlberger, J., Dalianis, H., Hassel, M., & Knutsson, O. (2001). Improving precision in information retrieval for Swedish using stemming. In *The 3th Nordic conference on computational linguistics NODALIDA'01* (pp.22-21), Uppsala, Sweden.

Chris, P.D. (1990). Another stemmer. ACM SIGIR.24(3), 61-56.

Chris, P.D. (1994). An evaluation method for stemming algorithms. In *the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (pp.50-42).

Croft, B.W., Xu, J. (1998). Corpus-based stemming using co-occurrence of word variants. *ACM Transactions on Information Systems*, 16(1), 81-61.

Dawson, J. (1974). Suffix removal and word conation. *Bulletin of the Association for Literary and Linguistic Computing*, 2(3), 47-33.

Deepika, S. (2012). Stemming Algorithms: A Comparative Study and their Analysis. *International Journal of Applied Information Systems* (IJAIS), 4(3), 12-7.

diBijal, D., & Sanket, S. (2014). Overview of Stemming Algorithm for Indian and Non-Indian Languages. *International Journal of Computer Sciences and Information Technologies (IJCSIT)*, 5(2), 1146-1144.

Ekmekcioglu, F.C., Lynch, M.F., & Willett, P. (1996). Stemming and N-Gram matching for term conation in Turkish texts. *Information Research News*, 7(1),6-2.

Frakes, W.B. (1992). Stemming Algorithm in Information Retrieval Data Structures and Algorithm ed. C. van Rijsbergen, Chapter 8: 139-132.

Frakes, W.B., & Fox, C.J. (2003). Strength and similarity of a x removal stemming algorithms. *SIGIR03*, 37,30-26

Frakes, W.B. (1984). Term Conation for Information Retrieval: In Research and Development. In Information Retrieval. ed. C. van Rijsbergen, New York: Cambridge University Press.

Funchun, P., Nawaas, A., Xin, L. and Yumao, L. (2007). Context sensitive stemming for web search. In *the 30th annual international ACM SIGIR conference on Research and development in IR* (pp.646-639).

Gadri, S., Moussaoui, A.O. (2015). Information Retrieval: A New Multilingual Stemmer Based on a Statistical Approach. In *3th International conference on Control, Engineering & Information Technology (CEIT'15)*, Tlemcen, Algeria, 25-27 may 2015

Greengrass, M., Robertson, A.M., Robyn, S., & Willett, P. (1996). Context sensitive stemming for web search in processing morphological variants in searches of Latin text. *Information research news;* 6(4), 5-2.

Gupta, V. (2014). Hindi Rule Based Stemmer for Nouns. *International Journal of Advanced Research in Computer Science and Software Engineering;* 4(1), 65-62.

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

Hafer, M., & Weiss, S. (1974). Word Segmentation by Letter Successor Varieties. *Information Storage and Retrieval*, 10, 371-86.

Harman, D. (1991). How effective is suffixing? *Journal of the American Society for Information*

Science, 42(1), 15-7.

Hmeidi, I., Al-Shalabi, R., Al-Taani, A.T., Najadat, H., & Al-Hazaimeh, S.A. (2010). A novel approach to the extraction of roots from Arabic words using bigrams. *Journal of the Association for Information Science and Technology*, 61(3),591-583.

Hull, D.A., Grefenstette, A. (1996). *A detailed analysis of English Stemming Algorithms*.

XEROX Technical Report on http://www.xrce.xerox.

Husain, M.S. (2012). An unsupervised approach to develop stemmer. *International Journal*

on Natural Language Computing IJNLC, 1(2),23-15.

Ionan, P. (2008). Web Document Classification and its Performance Evaluation. In *the 9th WSEAS*

*International Conference on evolutionary computing EC08 (*pp.353-348), Sofia, Bulgaria, May 2-4,

Jivani, A. (2011). A Comparative Study of Stemming Algorithms. *International Journal of Computer Technology and Applications IJCTA*, 2(6), 1938-1930.

Kanaan, G., Al-Shalabi, R., & Al-Kabi, M. (2005). New approach for Extracting quadrilateral Arabic roots. *Journal of Basic Science and Engineering*, 14(1),66-51.

Kasthuri, M., & Kumar, S.B.R. (2014). A comprehensive analyze of stemming algorithms for Indian and non-indian languages. *International Journal of Computer Engineering and Application IJCA*, 7(3), 8-1.

Kent, A., Berry, M., Luehrs, F.U., & Perry, J.W. (1955). Machine literature searching VIII.

Operational criteria for designing information retrieval systems. *American Documentation*, 6(2), 101-93.

Khodja, S. (1999). *Stemming Arabic Text*. Technical Report, Computing Department, Lancaster University.

Komkid, C., Narodom, K., Kittisak, K., & Netaya, K. (2012). Comparison of Feature Selection and Classification Algorithms for Restaurant Dataset Classification. In *the 11th conference on*

Latest Advances in Systems Science and Computational Intelligence (pp.134-129).

Krovetz, R. (1993). Viewing morphology as an inference process. In *the 16th annual international ACM SIGIR conference on Research and development in information retrieval*

*(*pp.202-191).

Larkey, L., & Connell, M.E. (2002). Arabic information retrieval at UMass in TREC-10. In *the 10th retrieval conference TREC-2001* (pp.570-562), Gaithersburg: NIST.

Lennon, M., Pierce, D.S., Tarry, B.D. and Willett, P. (1988). An evaluation of some conflation algorithms for information retrieval. *Document retrieval systems*: 105-99. P. Willett (Ed), Taylor Graham Publishing, London.

Levenstein, V.I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8), 710-707.

Lovins, J.B. (1968). Development of a stemming algorithm. *Mechanical Translation and*

computational Linguistics, 11(2), 31-22.

Majumder, M., Mitra, S., Parui, K., Kole, G., & Mitra, P. (20O7). YASS: Yet another Suffix Stripper. *ACM Transaction on Information Systems (TOIS)*, 25(4),5-6

Melucci, M., Orio, N. (2003). A novel method for stemmer generation based on hidden Markov models. In *the twelfth international conference on Information and knowledge management* (pp. 138-131).

Mesleh, A.W. (2007). SVM based Arabic Language Text Classification System: Feature Selection Comparative study. In *the 12th WSEAS International Conference on applied mathematics* (pp. 31-29.*)*, Cairo, Egypt,

Mishra, U., & Chandra, P. (2012). MAULIK: An Effective Stemmer for Hindi Language. *International Journal on Computer Science and Engineering (IJCSE)*, 4(5), 711-717.

Mohamad, A., Al-Shalabi, R., Kanaan, G., & Al-Nobani, A. (2012). Building An effective Rule-Based light stemmer for Arabic language to improve search effectiveness. *The International Arab Journal of Information Technology IAJIT*, 9(4),372-368.

Momani, M.; & Faraj, J. (2009). A novel algorithm to extract tri- literal Arabic roots, New Rules to Directly Extract Roots of Arabic Words. *International Conference on Computer Systems and Applications* (pp.315-309). IEEE Xplore Press.

Monz, C., & Rijke, M. (2001). Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. in *the 2nd workshop of Cross-language information retrieval*

and evaluation Form CLEF 2001 (pp.359-357), C. Peters Ed, Springer Verlag.

Moral, C., de Antoni, A., Imbert, R., & Ramirez, J. (2014). A survey of stemming algorithms in information retrieval. *Information Research*, 19(1), 605.

Moulinier, I., McCulloh, A., & Lund, E. (2001). Non-English monolingual retrieval. In Cross-language information retrieval and evaluation. In *the CLEF 2000 workshop* (pp.187-176), C. Peters Ed, Springer Verlag.

Mustafa, S.H., & Al-Radaideh, Q.A. (2004). Using n-grams for Arabic text searching. *Journal of the American Society for Information Science and Technology*; 55, 1007-1002.

Paik, J.H., Mitra, M., Swapan, K., & Jarvelin, P.K. (2011). GRAS: An effective and efficient stemming algorithm for information retrieval. *ACM Transaction on Information System (TOIS)*, 29(4), 24-20.

Porter, M.F. (2001). Snowball: A language for stemming Algorithm, October 2001, retrieved may, 2017.

Porter, M.F. (1980). An algorithm for suffix stripping. *Program*,14(3), 137-130.

Suprabhat, D., & Pabitra, M. (2011). A Rule-based Approach of Stemming for Inflectional and Derivational Words in Bengali. In *the IEEE Students' Technology Symposium* (pp.136-134),

jan 14-16, 2001, Kharagpur.

Tan, A.H., & Yu, P. (2000). A Comparative Study on Chinese Text Categorization Methods. *workshop on Text and Web Mining* (pp.35-24), Melbourne.

Thangarasu, M., & Manavalan, R. (2013). A Literature Review: Stemming Algorithms for

Indian Languages. *International Journal of Computer Trends and Technology (IJCTT)*, 4(8), 2584-2582.

Yousef, N., Aymen, A.E., Ashraf, O., & Hayel, K. (2014). An Improved Arabic Words Roots Extraction Method Using N-gram Technique. *Journal of Computer science JSC*, 10(4), 716.