Islamic University, Gaza, Palestine
Research and Graduate Affairs
Faculty of Engineering
Computer Engineering Department

Building an Arabic Word Stemmer for Textual Document Classification

بناء مجذر للكلمات العربية لتصنيف الملفات النصية

Mahmoud Eleyan Al Zaalan

Supervised by:
Dr. Mohammed Alhanjouri

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering
1435هـ - 2014 م

إقــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

Building an Arabic Word Stemmer for Textual Document Classification

بناء مجذر للكلمات العربية لتصنيف الملفات النصية

أقر بأن ما اشتملت عليه هذه الرسالة إنما هي نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وإن هذه الرسالة ككل، أو أي جزء منها لم يقدم من قبل لنيل درجة أو لقب علمي أو بحثي لدى أية مؤسسة تعليمية أو بحثية أخرى.

DECLARATION

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification

Student's name: **اسم الطالب: محمود عليان الزعلان**

Signature: **التوقيع:**

Date: **التاريخ: 2015/1/18م**

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza

مكتب نائب الرئيس للبحث العلمي والدراسات العليا هاتف داخلي 1150

الرقم.....ج س غ/35/..... Ref

التاريخ 2014/07/07 Date.....

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ محمود عليان عطية الزعلان لنيل درجة الماجستير في كلية الهندسة قسم هندسة الحاسوب وموضوعها:

بناء مجذر للكلمات العربية لتصنيف الملفات النصية

Building an Arabic Word Stemmer for Textual Document Classification

وبعد المناقشة التي تمت اليوم الاثنين 09 رمضان 1435هـ، الموافق 2014/07/07م الساعة الواحدة والنصف ظهراً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. محمد أحمد الحنجوري	مشرفاً ورئيساً	.....
أ.د. إبراهيم سليمان أبو هيبة	مناقشاً داخلياً	.....
د. إيهاب صلاح زقوت	مناقشاً خارجيًا	.....

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في كلية الهندسة/ قسم هندسة الحاسوب.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله ولزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق،،،

مساعد نائب الرئيس للبحث العلمي والدراسات العليا

أ.د. فؤاد علي العاجز

P.O. Box 108, Rimal, Gaza, Palestine fax: +970 (8) 286 0800 فاكس Tel: +970 (8) 286 0700 ص.ب 108 الرمال غزة فلسطين هاتف
public@iugaza.edu.ps www.iugaza.edu.ps

Dedication

To my father ...

To my mother ...

To my brothers and sisters ...

To My Friends...

To all who helped me, I dedicate this work.

ACKNOWLEDGMENTS

Firstly, I thank Almighty ALLAH for making this work possible. Then, there are a number of people to whom I am greatly indebted as without them this thesis might not have been written.

To Dr.Mohammed Alhanjouri for his guidance, support, and advice.

To my parents for providing me with the opportunity to be where I am. Without them, none of this would be even possible to do. You have always been around supporting and encouraging me.

To my brothers, sisters, and friends for their encouragement, input and constructive criticism, which are really priceless.

Contents

LIST OF ABBREVIATION

ASR	Automated Speech Recognition
BBC	British Broadcasting Corporation
CNN	Cable News Network
idf	Inverse Document Frequency
IR	Information Retrieval
K-NN	K Nearest Neighbors
LSA	Latent Semantic Analysis
Min-F	Minimum Term Frequency
NB	Naïve Bayes
Norm	Normalize data
OSAC	Open Source Arabic Corpus
SP_WOAL	Remove Suffixe then Prefix then Match with Pattern
SVM	Support Vector Machine
TC	Text Categorization
tf	Term Frequency

List of Figures

List of Tables

بناء مجذر للكلمات العربية لتصنيف الملفات النصية

محمود عليان الزعلان

الملخص

تقترح هذه الأطروحة خوارزمية جديدة لتجذير الكلمات تقوم بمعالجة مشاكل عدم الوضوح، والكلمات غير النظامية، وجموع التكسير الموجودة في خوارزميات التجذير الحالية، والتي تنقسم إلى قسمين، التجذير الخفيف والجذري.

تعتمد الخوارزمية المقترحة على إدخال قواعد جديدة للأنماط والتي تزيد كفاءة تحديد الكلمات. هذه الخوارزمية ستسهم في تعزيز كفاءة وسرعة استرجاع المعلومات ومحركات البحث. باستخدام هذه القواعد يمكن تحديد ما إذا كان تسلسل الزوائد هو جزء من الكلمة الاصلية أم لا. وبالتالي يمكن حل مشكلة عدم الوضوح.

تم تطوير أداة جديدة لإسترجاع المعلومات في اللغة العربية باستخدام لغة البرمجة JAVA مع JDK 1.6، وتمتلك هذه الأداة العديد من الخيارات، فهي تتيح للمستخدم تحميل أي مجموعة بيانات و الإختيار من أحد المجذعات الموجودة و الاختيار من بين ثمانية خطوات في مرحلة التطبيع وتحديد مجموعة من الثوابت مثل "البادئات، اللواحق، الكلمات المراد حذفها" و تصنيف النص و إجراء المقارنات بين المجذعات وإستخراج الأشكال التوضيحية التي تبين هذه المقارنات. تستخدم الأداة الجديدة في اختبار المجذع المقترح، وتظهر النتائج التي تم استخراجها باستخدام كل من CNN و BBC و OSAC ان المجذع المقترح يزيد دقة تصنيف النصوص إلى متوسط قدره 91.7٪ وهوأفضل من استخدام مجذع Light 10 او Khoja اللذان يحققان متوسط دقة 90.2٪ و 89.17٪على التوالي.

Building an Arabic Word Stemmer for Textual Document Classification

Mahmoud Aleyan Alzaalan

ABSTRACT

This thesis proposes a new stemming algorithm that addresses the ambiguity, irregular words and broken plural problems in current stemming algorithms, which are divided to two approaches, the root stemming and the light stemming.

The proposed algorithm will depend on introducing new rules of patterns which increase efficiency of identifying words. Such algorithm will contribute to enhanced efficiency and speed of information retrieval and search engines. By using these rules, it can determine whether the sequence of affixes is a part of the real word or not. Thus the ambiguity problem can be solved.

A new Arabic IR tool has been developed which has many options using java programming language with JDK 1.6; it allows user to load any data set, choose from any included stemmers, choose from the eight normalization steps, define the set of constants like “prefixes, suffixes, stopwords”, text classification, make comparisons between stemmers and extract charts that show these comparisons. The new tool used to test the proposed stemmer and the results which has been derived using CNN, BBC and OSAC corpora show that the proposed stemmer increases accuracy of text classification to an average of 91.7% which is better than using Light 10 or Khoja which achieve average accuracy of 90.2 % and 89.17% respectively.

Keywords:

Arabic Stemming, Root Stemming, Text classifications, Naïve Bayes Multinomial, K-NN.

Chapter 1. Introduction

Arabic information retrieval has become increasingly important, due to the increased availability of documents in digital form and the need to access them in flexible ways. The need of perfect tools and techniques that assist users in finding and extracting relevant information from large data is high [1].

1.1 Information retrieval

Information retrieval (IR) is the art and science of searching for information in documents, searching for documents themselves, searching for metadata which describe documents, or searching within databases, whether relational stand alone databases or hypertext networked databases such as the Internet or intranets, for text, sound, images or data. It is the art and science of retrieving from a collection of items that serves the user purpose. The main purpose is to retrieve what is useful while leaving behind what is not [2].

Traditionally, IR has concentrated on finding whole documents consisting of written text; most IR researches focuses more specifically on text retrieval. But there are many other interesting areas [1]:

- Speech retrieval, which deals with speech, often transcribed manually or (with errors) by automated speech recognition (ASR).
- Cross language retrieval, which uses a query in one language (say English) and finds documents in other languages (say Arabic)
- Question answering IR systems, which retrieve answers from a body of text.
- Image retrieval, which finds images on a theme or images that contain a given shape or color.

To increase the efficiency of information retrieval we use stemming techniques; stemmers are basic elements in query systems, classifications, search engines and information retrieval systems (IRS). Stemming for IR is a computational process by which suffixes and prefixes are removed from a textual word to extract its basic form. The basic form produced does not have to be the root itself. Instead, the stem is said to be the least common denominator for the morphological variants [3].

Stemming has two basic types: First, root stemming in which each word returns to its basic root by removing all additional infixes, the second is light stemming which refers to a process of stripping off a small set of prefixes and/or suffixes, without trying to deal with infixes, or recognize patterns and find roots [4].

The importance of word stemming for information retrieval and computational linguistics was pointed out by Lennon et al. [5], the notion is thought to be useful for two reasons; firstly, it reduces the total number of distinct terms present with a consequent reduction in dictionary size and updating problems. Secondly, similar words generally have similar meanings and thus retrieval effectiveness may be increased. From an application perspective, stemming has been seen useful in two ways [6]. In the first, roots extracted can be used in text compression, text searching, spell checking, dictionary lookup, and text analysis. In the second, affixes recognized can be used in determining the grammatical structure of the word, which is important to linguists.

The effect of term stemming on the performance effectiveness of information retrieval has been the subject of several investigations. Most notably of these investigations are those reported by [5] [7] [8]. The general indication coming out of most studies is that stemming improves retrieval performance, and improves recall more than precision [9].

1.2 Arabic Language

Arabic language is one of the most complex languages, in both its spoken and written forms. However, it is also one of the most common languages in the world as it is spoken by more than 400 million people as a first language and by 250 million as a second language [10]. Arabic Language belongs to the Semitic language family.

Arabic alphabet consists of 28 letters that structure the words; words are divided into three parts of speech: noun, verb, and particle. Nouns and verbs are derived from a closed set of around 11,311 roots distributed as follows [11]:

- 115 two character roots (no derivation from them).
- 7198 three character roots.
- 3739 four character roots.

- 259 five characters roots.

These roots can be joined with several infixes to generate more patterns of words [12], for example several forms can be derived from the pattern "فعل" of the morpheme "صنع" , the form "مصنع" can be found by adding the letter "م" to the morpheme "صنع".

The Arabic script has numerous diacritics (Damma, Fathah, Kasra, Shaddah) which decide how a word should be pronounced. Arabic has two genders (feminine and masculine), three cardinalities (singular, dual and plural), three grammatical cases (nominative, genitive and accusative), and two tenses (perfect and imperfect). Arabic nouns are formed differently depending on the noun gender, cardinality, and grammatical case [13].

1.3 Complexity of Arabic Language

Arabic is considered as one of the highly inflectional languages with complex morphology and considered as challenging language for a number of reasons [14] [15] [16]:

- Morphological variation and the agglutination phenomenon, Letters change forms according to their position in the word (beginning, middle, end and separate) as shown in Table 1.1.

Table 1.1 Different shapes of letter "ع ″ depending of its position in the word

Beginning	Middle	End	Separate
عـ	ـعـ	ـع	ع

- Arabic plurals are formed more irregularly than in English; depending on the root and the singular form of the word, the plural form might be produced by the addition of suffixes, prefixes or infixes, or by a complete reformulation of the word.
- There is no space between a word and its prefix, postfix and pronoun; that makes the boundary between the word and the preposition invisible.
- It is common to find many Arabic words that have different pronunciations and meanings but share the same written form (homonyms), making finding the appropriate semantic occurrence of a given word a problem, for example

the word "ذهب"may refer to the word "gold" or "went" depending on the diacritics.

§ Many words can refer to the same meaning that may lead to information mismatch in search process, example "ظهر–برز–بان".

§ Arabic words may change according to their case modes (nominative, accusative or genitive); "مفاوضون–مفاوضين"

1.4 Thesis Motivation and Objective

Although a lot of stemmers have been applied, most of these stemmers still suffer from many problems like the absence of morphological rule, which helps to determine the correct affixes in the word, the irregular words, the broken plurals and the use of full root dictionary to extract the root. The main objective of this thesis is to propose a system for Arabic stemming that solves all of the above mentioned problems.

1.5 Thesis Contribution

This thesis will contribute with the following:

- Developing the proposed stemmer depending on rule based techniques, show the effects of normalization and tokenization into stemming techniques.
- Automatically detect the irregular words "non Arabic words" by applying the rules, so any word that does not match rule will be considered as irregular and returned without stemming.
- Adding the proposed stemmer to one of the most famous IR platforms "WEKA".
- Developing new Arabic information retrieval tool with graphical user interface that allow user to analyze data and compare between several results, or gather between several techniques.
- Allow developers to add or modify to the new tool as it is an open source environment.

1.6 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2: Introduces the related work and the text categorization process as example for testing the stemmer.

Chapter 3: Describes the methodology including the proposed stemmer and the new Arabic IR tool.

Chapter 4: Will show the results of the work.

Chapter 5: The conclusion of the research, which will summarize the research.

Chapter 2. Literature Review

2.1 Stemmers' Algorithms

Stemming is the process of converting several forms of a word into a single representation; the stem does not always be the original word, but it must have a meaning of the word. In IR the stemming is used to avoid mismatches between the words which derived from the same root such as "المدرسين", "المدرس", "المدرسان".

Many stemming methods have been developed in English, other European languages, and Asian languages such as Chinese. These algorithms are used to increase the performance of IR systems from 10 to 50 times [17]. However, research studies on stemming for Arabic language have increased over the last years and most of these studies have used the morphological meaning to extract the root or the stem "light stemming".

Figure 2.1 shows various approaches that can be used in stemming. There are three approaches: Table lookup method, Affix Removal Method and n-gram Method.

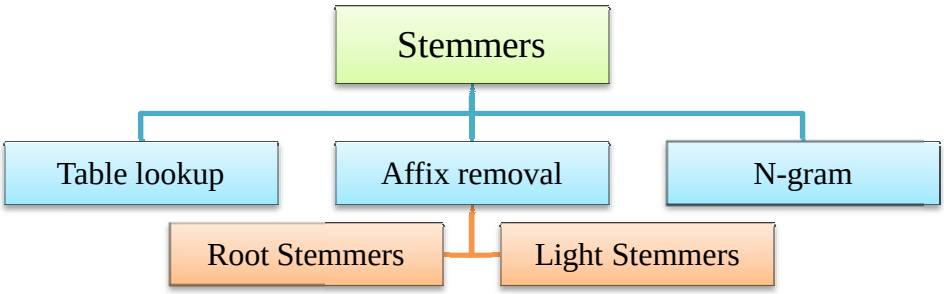


Figure 2.1 Stemming Approaches

Stemmer accuracy can be computed by retrieval effectiveness that is usually measured by recall, precision and time. Accuracy of stemmer can be affected by overstemming and under stemming. Overstemming means that too much of a term is removed, while under stemming is the removal of too little of a term.

2.1.1 Table lookup

Word and its stem stored in a table, stemming is then done by looking up in the table. Hash table can be used to fasten the search process but this method still suffers from the huge data needed to be stored and the continuous refreshment of the table contents.

2.1.2 Affix removal

This method depends on removing the suffix, prefix and/or infixes from the words so as to return them into a common stem form "The root or other pattern". Affix removal method can be divided to two approaches root and light stemmers.

1) **Root Stemmers**

Khoja [6] developed a root stemmer depending on morphological patterns, the stemmer firstly removes the infixes, suffixes and prefixes from the word and then matches the result against set of patterns in order to extract the root. Then it checks against set of predefined roots to detect if it is a true root or not. The stemmer uses several static data like stopwords, punctuations and diacritic. The weakness of this algorithm is that the root list needs to be continuously updated to ensure that new words are correctly stemmed.

Al-Shalabi and Evens [18] developed a system for extracting the roots of Arabic words. It first removes the longest prefix that precedes the first root letter in the input word. It then checks for the root in the new word formed by removing the prefix. Typically, the root would be within the first four or five letters.

Al-Shalabi et al. [19] developed a root extraction algorithm which does not use any dictionary. It depends on assigning weights for a word's letters multiplied by the letter's position, consonants were assigned a weight of zero and different weights were assigned to the letters grouped in the word "سألتمونيها".The algorithm selects the letters with the lowest weights as root letters.

Taghva et al. [20] shared many features with the Khoja stemmer. However, the main difference is that it does not use root dictionary. Also, if a root is not found, the stemmer returns normalized form, rather than returning the original unmodified word. The algorithm firstly remove prefixes and suffixes of two and three letters length from the word and then matches the remaining word with a set of predefined patterns, If a match is found, extract the relevant stem and return. If not, then try to remove one additional prefixes or suffixes and rematch the word.

Boubas et al. [21] used genetic algorithms and pattern matching to generate a morphological analyzer for Arabic verbs. GENESTEM begin by developing general

verbs patterns and then applying these patterns to derive morphological rules. The algorithm defines 3089 patterns that can be applied to the verbs of length three then matches the words against these patterns, when pattern matched the extra characters will be removed and only the root will be kept.

Kanaan [22] stemmer utilized an important morphological aspect of the Arabic language. The algorithm examines the word letter by letter starting from the end of the word, the letter is checked to determine if it is additional letter or not, each letter that found into the list [أ, ت,م, و, ن, ي, ا, ء, ئ] will be considered as additional, while any other letters will considered as original. For each additional letter a set of rules has been defined to decide whether to delete the letter or add it to a list. These rules depend on what precedes the letter and what follows it. Finally, the list will be resorted according to the original appearance of each letter in the original word. The algorithm has been tested on a corpus of 242 abstracts of Arabic documents and achieved an accuracy rate of 97.6%.

Yaseen and Hmeidi [23] developed the Word Substring Stemming Algorithm that does not remove affixes during the extraction process. The algorithm is based on producing the set of all substrings of an Arabic word, and uses the Arabic roots file, the Arabic patterns file and a concrete set of rules to extract correct roots from substrings. The experiments have shown that the proposed approach accuracy is 83.9%. Furthermore, the algorithm seems to be suffering from the same Khoja [6] weakness, which is the update of roots file.

2) Light Stemmers

The objective of light stemming is to find the representative indexing form of a word by the application of truncation of affixes. The main goal of light stemming is to return the word meaning intact, and so improve the retrieval performance. Light stemming is mentioned by some authors, but till now there is no standard algorithm for Arabic light stemming, all trials in this field were a set of rules to strip off a small set of suffixes and prefixes, also there is no definite list of these strippable affixes.

Larkey [4] classified affixes to four kinds: antefixes, prefixes, suffixes and postfixes that can be attached to word.

Table 2.1: An agglutinated form of an Arabic word meaning

Antefix	Prefix	Core	Suffix	Postfix
ل	ي	ناقش	ون	هم

From the table above, Larkey said that if we can remove all affixes from the word then we will get a stemmed word that have a meaning and so we will improve the search effectiveness. The weakness of Larkey is that it removes affixes predefined in the list without checking if it is a stem and in some cases, truncates it from the word and produces an erroneous stem.

Aljlayl and Frieder (Al-Stem) [24] developed a light stemmer used for his own information retrieval researches. The stemmer defines a set of most frequent suffixes and prefixes that occurs in the words to be removed, the stemmer removes prefixes while word length is greater than three characters, or there is no prefixes found in the word [the longest prefixes will be removed first], after that if the word length still greater than three then the suffixes will be removed with the same conditions of prefixes. The disadvantage of this technique is the blind removal of affixes from the beginnings and ends of the words as it is done without any prior knowledge (linguistics rule).

Al Ameed et al. [25] study of Al-stem [24], Larkey [4] and other stemmers and enhance the performance of these stemmers in two ways. First enhancement is done by adding new affixes. The second way is by reordering the algorithm iterations. The stemmer works as follows, firstly remove the prefix "ال"from the beginning of the word and then remove all suffixes from the end, and finally the stemmer will remove the prefixes starting from the longest.

Alhanini and Aziz [26] developed a stemmer based on light stemming and dictionary-match approach. The stemmer aims to solve the problem of irregular words that cannot be stemmed correctly by using affixes removal, so it firstly searches a pre defined dictionary and if it is not found it applies affixes removal process. The stemmer has been tested against Arabic corpus and achieved average accuracy equal to 96.2 %.

Nwesri et al. 2005 [27] developed a stemmer that only removes conjunction and preposition affixes without taking care of other affixes, he thought that removing other affixes will affect the meaning of the word.

Nwesri et al. 2007 [28] developed a stemmer called (Restrict Stemmer) that its main goal is to validate Arabic stemmed words by using Microsoft Office 2003 Arabic spellchecker to ensure that it is a correct one. The disadvantage of this technique is that its rules do not guarantee a hundred percent correctness. It needs a lexicon which contains all the forms of all the words in Arabic language which is very difficult to obtain.

Delekh and Bhloul [29] developed a new stemmer that is a combination between three Arabic stemming techniques [affixes removal, lookup and morphological analysis]. They have developed five different stemming methods by making combination between the previous three techniques and compared their results in information retrieval. The main idea of these five stemmers depends on which is removed first, suffixes then prefixes or prefixes then suffixes, or matching against word before removing affixes or after and so on. The results show that prefix-suffix match achieves the highest accuracy.

Tashaphyne [30] developed a light stemmer that depends on matching the word against list of predefined rules. The algorithm at first normalizes the word by removing diacritics, prefixes and suffixes, then compares the remaining word with a predefined list of rules. The algorithm also uses a new set of prefixes, which contains prefixes of lengths one to seven and a new suffixes list. It also provides an open source library that allows user to find the stemmed word, normalized word and also allows user to change the stemmer behavior.

Kadri and Nie [31] developed a stemmer that considers the Arabic word consists of five parts, their order is; antefixes, prefixes, stem, suffixes and postfixes. The first part, which is the antefixes, is the prepositions and conjunctions. The prefixes are the conjugations person of verbs. The suffixes are termination of conjugation and number marks of nouns. The postfixes are the pronouns added to the end of the word. The stemmer truncates a word from its two ends. The decision to truncate a segment of a word or not, is made according to rules and statistics based on the corpus. After removing the predefined affixes from the word, then the remaining stem is compared

with a list of stems, which have been predefined according to the rules and returns the matching stem word.

1- Normalize word
   - Replace "إ" ،"أ" and "آ" by alif bar "ا"
   - Replace "ى" by "ي" at the end of the words.
   - Replace "ة" by "ه" at the end of the words.
   - Replace the sequence "يء" by "ى"
   - Remove the tatweel character "-", used for aesthetic writing in the Arabic texts.
   - Removing the shedda "ّ" and the diacritics.
2- Remove Prefixes
3- Remove Suffixes

Figure 2.2: steps of Arabic light stemmer

Figure 2.2 defines the steps of light stemmer algorithm used by most light stemmer techniques like Larky [4], Kadri [31] and Chin [32]. The main difference is that each algorithm tries to define a set of prefixes and suffixes, and defines a set of rules that manage the removal process.

Mustafa [33] depends on studying the merits of light stemming for Arabic data and presents a simple light stemming strategy that has been developed on the basis of an analysis of actual occurrence of suffixes and prefixes in real texts, the study indicates that only a few of the prefixes and suffixes have an impact on the correctness of stems generated.

Nehar, et al. [34] developed a new stemming approach, which is used in the context of Arabic text classification. It is based on the use of transducers for both words stemming and distance measuring between documents. First, the transducer for stemming is built by mean the Arabic patterns. Second, transducers will be also used to calculate distances.

2.1.3 N-Gram

The N-gram measures the similarity of two words according to the structures of characters of these words. Two words are considered similar if they have in common several substring of N characters, this is done by calculating a coefficient on these two words. N-gram does not need knowledge of the language and does not need to have any predefined sets of rules or tables.

N-grams may be based on the stemmed word or the original word, one which is based on the stemmed word is better than the one based on original word, because the original word based N-gram could have prefixes and suffixes which make more mistakes in the similarity between the document and query.

W. Adamson George and J. Boreham (1974) [35] have developed the first classifier based on bigrams (2-gram) to compute the similarity between pairs of character strings, as described in Table 2.2 and Figure 2.3.

Table 2.2: bigram (2-gram) for two words

2-gram							الكلمة
ام	حا	دح	زد	از	لا	ال	الازدحام
2-gram							**الكلمة**
ام	حا	دح	زد	از			ازدحام

Table 2.2 shows the unique bigrams for two words"الازدحام" and "ازدحام". The first word consists of seven unique bigrams and the second consist of five.

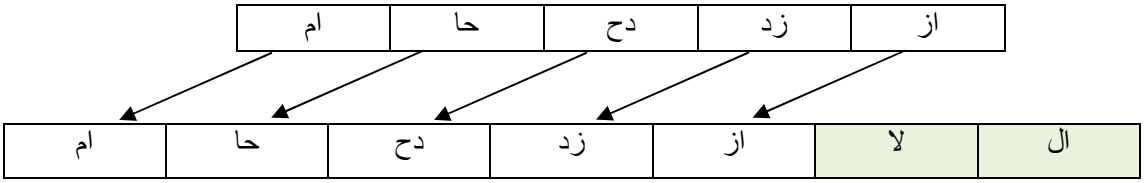


Figure 2.3 : bigram similarity measure between two words الازدحام and ازدحام

F. Ahmed and A. Nrnberger (2007) [36] have developed the n-gram model that counts the number of similar n-grams between two words starting from bigram and continuing till there is no match found.

2.2 Comparative Studies

Many research studies have been done to compare between different stemming algorithms. These studies were based on different criteria for measuring the accuracy of the algorithm including compute the recall, precision, time, comparing between the main ideas, list of prefixes and suffixes used and so on. The studies demonstrated that light stemmer is better for finding words related together than root stemmers as the last one affect the word meaning. The studies also indicated that Khoja and Larkey-Light10 are the best stemmers for root and light stemming respectively.

Froud, Lachkar and Ouatik [37] compared between root stemming and light stemming techniques for measuring the similarity between Arabic words with Latent Semantic Analysis (LSA) model. They experimented using two datasets both are collected from Saudi Press Agency, the first one includes 255 files divided into three classes and the second consist of only one class with 257 files. The results show that using cosine similarity will be more efficient than using Euclidean while the overall results show that the light stemming outperformed the root stemming approach because the last one affects word meaning.

Saaed and Ashour [38] studied the effects of using stemming on text classification accuracy. Their study used two stemmers: Khoja [6] as a root stemmer and Light 10 [4] as a light stemmer. They also studied the effects of preprocessing time, distance measurement and weighting techniques. The stemmers were tested against seven datasets including OSAC, CNN, BBC and others, and used the most famous classification techniques like SVM, NB and k-means. The results show that the text preprocessing has great effects on stemmer's results, and using of root stemmers has slight better average accuracy than light stemmers but with more execution time. Finally, they recommended using light stemming as it is more proper than root stemming from linguistics and semantic point of view and it has the least preprocessing time.

Darwish's [39] compared Al-Stem with five attempts for enhancement proposed by Al-Ameed et al. The list of affixes to be removed included more prefixes and suffixes than those used in Al-Stem. The researcher claimed that Al-Stem stemmer provided better accepted (meaningful) outcomes with up to 30-40% more than those reported by the TREC-2002 stemmer.

Said et al. [40] found that light stemming is better than root stemming; the study was done using Al-Stem and Sebawai root extractor. The study was performed using four feature scoring methods and different threshold values. Two datasets were used in this study; namely Alj-News, and Alj-Mgz datasets. The results show that using light stemmer with a good-performing feature selection method such as MI or IG enhances the performance.

Bsoul and Masnizah [41] evaluated the impact of the five measures [Cosine similarity, Jaccard coefficient, Pearson correlation, Euclidean distance and Averaged

Kullback-Leibler divergence] for document clustering with two types Taghva root stemmer and without stemming. They used a dataset consisted of 4 categories namely art, economics, politics and sport articles, and each contains documents taken from Al-salemi and Aziz [42] 1680 documents where used in testing the dataset. They concluded that the method of Taghva is proved to be better than without stemming, which use a five similarities/distance measures for document clustering. That is because without stemming has under stemming error in which some terms that should be stemmed to one root are not, which leads to creating similarities among the unrelated documents containing the same roots for different words.

De Roeck & Al-Fares [43] found that light stemming gives better results than root stemming. They noticed that root stemming may lead to under stemming problem. The word "منظمات", which comes from the root "نظم", is stemmed by a root stemmer into: "ظمأ" that does affect the classification by matching unrelated documents to each other. They said that by using light stemming the word will be stemmed to the true original pattern "منظم".

Kardi and Nie [31] have demonstrated that linguistic-based stemming using a 3-gram root can provide better retrieval results than light stemming. The linguistic approach used was similar to that proposed by Khoja. To select an acceptable root, they made use of the affix statistics provided by the TREC collection. As of the light stemmer, they identified 16 prefixes and 17 suffixes that should be removed by the stemmer.

El-Disooqi, Arafa and Darwish [44] compared between nine light stemmers: Al-Stem, Aljlayl, Light8, Berkeley Light Stemmer, Light10, SP_WOAL Light Stemmer (Al Alameed et al), Restrict Stemmer, linguistic-based stemmer (Kadri & Nie) and Elbeltagi stemmer. The comparison was done in terms of the main idea behind the stemmer build, prefixes and suffixes they remove, the basis of choosing the affixes, algorithm they use to remove the affixes, IR performance, precision and recall and finally the limitation of the stemmer. The results show that the Light10 stemmer outperformed the other stemmers in non-expanded experiments and Aljlayl outperform them in case of expansion. Aljlayl and Al-Stem experiments show that using different stemming algorithms for removing affixes even with the same affixes list produce different results.

Chapter 3. Background

3.1 Selected Stemmers

We mentioned in the previous section that using stemmers have great effects on the accuracy of information retrieval, and the studies refer that the best root and light stemmers to use are Khoja [6] and Larkey – Light 10 [4] respectively.

In this section we will discuss these two techniques as an example of root and light stemmers and to understand the basic idea of root/light stemming. The techniques will be also implemented in the new Arabic IR tool presented in this thesis and it will be compared with the proposed stemmer.

3.1.1 Shereen Khoja Stemmer:

Khoja's stemmer removes the longest suffix and the longest prefix, it then matches the remaining word with verbal and noun patterns, to extract the root. The stemmer makes use of several linguistic data files such as a list of all diacritic characters, punctuation characters, definite articles, and 168 stop words. The weaknesses of Khoja stemmer is that:

- Some words do not have roots in Root dictionary so it needs to be updated to include all newly Arabic roots.
- Blindly removal of suffixes and prefixes may lead to removal of original letters from the word, which will lead to wrong matching of the root.
- If the root contains a weak letter (i.e. alif الف, wawواو or yah ياء), the form of this letter may change during derivation, for example "منظمات" will be stemmed to "ظما".

The Khoja stemmer algorithm steps are described in Figure 3.1

1- Normalize word
   - Remove diacritics
   - Remove stopwords, punctuation, and numbers
   - Remove the tatweel character "ـ", used for aesthetic writing in the Arabic texts.
   - Remove definite article "ال" and conjunction "و".
   - Replace "أ", "إ" and "آ" by "ا".
2- Remove Prefixes.
3- Remove Suffixes.
4- Match result against a list of patterns if a match is found; extract the characters in the pattern representing the root.
5- Match the extracted root against a list of known valid roots.
6- Replace weak letters"ا", "و", "ي" with "و".
7- Two letter roots are checked to see if they should contain a double character. If so, character is added to the root.

Figure 3.1: The Khoja stemmer algorithm steps

3.1.2 Larkey-Light 10:

Larkey [4] used heuristic as a strategy for developing his stemmer. The stemmer removes the following prefixes: "ال, بال,وال, و, لل، فال، كال، " and it removes the following suffixes: "ي, ه, يه, ين, ون, ات, ان, ها". Larkey only removes definite articles. The stemmer does not remove any Arabic prefixes from words. The main basic steps of Larkey has been listed in Figure 3.2

1- Normalize word
   - Replace "أ", "إ" and "آ" by alif bar "ا"
   - Replace "ى" by "ي" at the end of the words.
   - Replace "ة" by "ه" at the end of the words.
   - Replace the sequence "يء" by "ى"
   - Remove the tatweel character "-", used for aesthetic writing in the Arabic texts.
   - Remove the shedda "ّ" and the diacritics.
2- Remove Stopwords
3- Remove و if the remainder of the word is 3 or more characters long.
4- Remove Prefixes - definite articles if this leaves 2 or more characters.
5- Remove Suffixes, if this leaves 2 or more characters.

Figure 3.2: The main basic steps of Larkey

3.2 Text Classifications

Text classification is the task of assigning predefined categories to free-text documents. It can provide conceptual views of document collections and has important applications in the real world. The first step of text categorization is to convert documents which are strings to vectors that represent these documents [45].

Information retrieval studies found that word stemming acts well in text classification as each word represents a feature and its value will be the number of occurrences of this word in the document. Using stemmers will lead to reduce the number of features by converting many forms of words to its original form [46].

Text Classification Process can be divided into three phases: text preprocessing, term weighting and classification, as in Figure 3.3.

Figure 3.3: Text Classification Process

The text classification problem is composed of several sub problems, such as the document indexing, the weighting assignment, document clustering, dimensionality reduction, threshold determination and the type of classifiers [45] [47]. Several methods have been used for text classification such as: Naïve Bayes (NB) [48] [49], K Nearest Neighbor (K-NN) [50] [51] [52].

3.2.1 Text preprocessing

In the preprocessing step, the documents should be transformed into a representation suitable for applying the learning algorithms. The most widely used method for document representation is the vector space model introduced by Gerard Salton [Gerard Salton et al, 1975] [53].

In this model, each document is represented as a vector d. Each dimension in the vector d stands for a distinct term (word) in the term space of the document collection.

Text preprocessing can be applied by applying tokenization, normalization, stopwords removal and finally applying the stemmer algorithm.

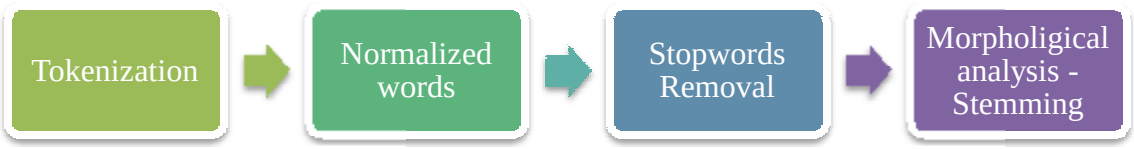


Figure 3.4: preprocessing steps

Tokenization process is the process of converting the document to individual words. Normalization is the process of removing diacritics, punctuations, numbers and any other unnecessary letters from the tokenized word.

The stopwords removal is the process of removing those words such as pronouns, prepositions and conjunctions that are used to provide structure in the language rather than content and carry little meaning. Keeping those words can affect the classification process as they have a very high frequency and tend to diminish the impact of frequency differences among less common words, affecting the weighting process. The process will also reduce the number of features and so increase the performance of the classifier; about 30% to 50% of the original words can represent stopwords [54].

3.2.2 Term Weighting

After text preprocessing each document from the collection of documents [Doc_1, Doc_2… Doc_n] is represented as a vector d. Each dimension in the vector d stands for a distinct term (word) in the term space of the document collection [$Term_{11}$, $Term_{12}$…. $Term_{1t}$]. Then the collection can be represented in a matrix form as shown in Figure 3.5:

	$\boldsymbol{Term_1}$	$\boldsymbol{Term_2}$	…	$\boldsymbol{Term_t}$
$\boldsymbol{Doc_1}$	t_{11}	t_{12}	…	t_{1t}
$\boldsymbol{Doc_2}$	t_{21}	t_{22}	…	t_{2t}
…	…	…	…	…
$\boldsymbol{Doc_n}$	t_{n1}	t_{n2}	…	t_{nt}

Figure 3.5: Weight Matrix of Vector Space Model

The term T vector will consist of all unique words that appear in each document of the collection, so the matrix will be sparse matrix as every word does not normally appear in each document.

There are several ways of determining the weight t_{nt} of word t in document n, but most of the approaches are based on two empirical observations regarding text [55]:

- The more times a word occurs in a document, the more relevant it is to the topic of the document.
- The more times the word occurs throughout all documents in the collection, the more poorly it discriminates between documents.

1) Boolean Weighting

The term t_{nt} value is considered as one if the word t appears in the document n, otherwise the value will equal zero.

2) Term Frequency Weighting

The count of appearances of each word t in the document n will be considered as the value of t_{nt} , if the word does not appear in the document then the value will equal zero.

3) Term Frequency Inverse Document Frequency weighting (tf-idf)

The previous two methods do not take into account the frequency of the word throughout all documents in the collection. tf-idf (term frequency-inverse document frequency) weighting assigns the weight to word t in document n in proportion to the number of occurrences of the word in the document, and in inverse proportion to the number of documents in the collection for which the word occurs at least once. The tf-idf weight can be represented by the following function:

$$t_{nt} = tf * idf \qquad \textbf{3.1}$$

Where tf is the number of appearances of word t into document n, and idf is the average between total number of documents and the documents that the word t appears in.

$$idf = \log \frac{N}{n_t} \qquad \textbf{3.2}$$

N: total number of documents.

n_t: number of documents where the word t appears in.

3.2.3 Classification

Classification is the process of building a set of models that can correctly predict the class of different objects. The derived model is built depending on the training data and after it has been built, it will be used to assign labels to new documents.

There are two different ways to build a classifier:

- Parametric: According to this approach, training data is used to estimate parameters of a distribution or discrimination function on the training set. The main example of this approach is the probabilistic Naive Bayes classifier.
- Non-parametric: These classifiers base classification on the training set itself. This approach may be further subdivided in two categories:
    - Example-based: According to this approach, the document d to be categorized is compared against the training set of documents. The document is assigned to the class of the most similar training documents. Example of this approach is k-Nearest Neighbor (K-NN) classifier.
    - Profile-based: In this approach, a profile (or linear classifier) for the category, in the form of a vector of weighted terms, is extracted from the training documents pre-categorized under c_i. The profile is then used as a training data against the document d to be categorized. Example of this approach is Support Vector Machines (SVM).

The most familiar classifications methods which will be used in this work to classify the documents and measure the performance of the stemmer will be presented.

1) Naive Bayes Multinomial

Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. Naive Bayes Multinomial is a specialized version of Naive Bayesian that is designed more for text documents. Whereas simple Naive Bayes might model a document with the presence and absence of particular words, Multinomial Naive Bayes explicitly models the word counts and adjusts the underlying calculations to achieve better accuracy. (McCallum and Nigam, 1998) [56].

Naïve Bayes estimates the probability that an instance x belongs to class y as:

$$P(Ci|dj) = \frac{P(dj|Ci)p(Ci)}{P(dj)} \quad \textbf{3.3}$$

The posterior probability of each category c_i given the test document d_j, i.e. $P(C_i \mid d_j)$, is calculated and the category with the highest probability is assigned to d_j. In order to calculate $P(C_i \mid d_j)$, $P(C_i)$ and $P(d_j \mid C_i)$ have to be estimated from the training set of documents. Note that $P(d_j)$ is same for each category so we can eliminate it from the computation. The category prior probability, $P(c_i)$, can be estimated as follows:

$$P(C_i) = \frac{N_i}{N} \quad \textbf{3.4}$$

Where N_i represents the number of documents belong to class I, while N is the total number of documents.

The Naive Bayes Multinomial assumption is that the probability of each term event is independent of term's context, position in the document, and length of the document. So, each document d_j is drawn from a multinomial distribution of terms with number of independent trials equal to the length of d_j. The probability of a document d_j given its category C_i can be approximated as:

$$P(d_j|C_i) \cong \prod_{k=1}^{|d_j|} P(t_k|C_i) \quad \textbf{3.5}$$

2) K-means algorithm

K-means is one of the most widely used partition-based clustering algorithms in practice. It is simple, easy, understandable, scalable, and can be adapted to deal with streaming data and very large datasets [57]. K-means algorithm divides a dataset X into k disjoint clusters based on the dissimilarities between data objects and cluster centroids. Let $\bar{\mu}_i$ be the centroid of cluster C_i and the distances between X_j that belong to C_i and $\bar{\mu}_i$ is equal to $d(Xj, \bar{\mu}_i)$. Then, the objective function minimized by K-means is given by:

$$\min_{\bar{\mu}_1,..,\bar{\mu}_k} E = \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, \bar{\mu}_i) \quad \textbf{3.6}$$

Where ‘d’ is one of distance function. Typically d is chosen as the Euclidean or Manhattan distance.

The Euclidean distance between points X and Y is the length of the line segment connecting them ($\overline{X\,Y}$). If X and Y are n-dimensional vectors where X= (x_1, x_2,..., x_n) and Y = (y_1, y_2,..., y_n), then the Euclidean distance from X to Y, or from Y to X is given by:

$$\begin{Bmatrix} d(X,Y) \\ d(Y,X) \end{Bmatrix} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \qquad \textbf{3.7}$$

The Manhattan distance between two points measured along axes at right angles where distance that would be traveled to get from one data point to the other if a grid-like path is followed. In a plane with X at (x_1, x_2) and Y at (y_2, y_2), it is $|x_1 - y_1| + |x_2 - y_2|$. The Manhattan distance between two n-dimensional vectors is the sum of the differences of their corresponding components.

$$d(X,Y) = \sum_{i=1}^{n} |x_i - y_i| \qquad \textbf{3.8}$$

Where n is the number of variables, and Xi and Yi are the values of the *i*th variable, at points X and Y respectively.

Usually the selection process between the two methods of calculating the distance is left to the user based on the nature of the data. Figure 3.6 shows the difference between using Euclidean and Manhattan distance to calculating the distance between two points in two-dimensional space.

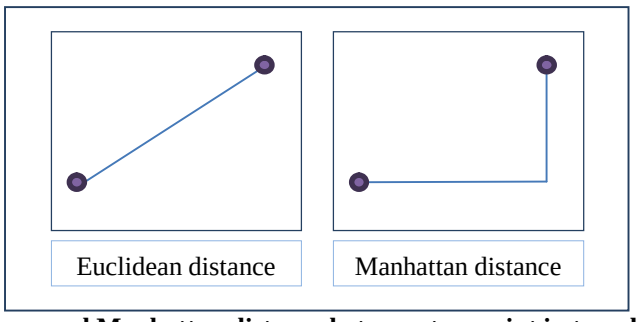


Figure 3.6: Euclidean and Manhattan distance between two point in tow-dimensional space.

K-means algorithm working process summarized as follows:

1. Determine the number of clusters (k parameters in k-means).

2. K-means selects randomly k cluster centroids.
3. Assign object to clusters based on distance function.
4. When all objects have been assigned, Re-compute new cluster centroids by averaging the observations assigned to a cluster.
5. Repeat steps 3 and 4 until convergence criterion is satisfied.

Chapter 4. Methodology and Design

Although a number of attempts had been made to develop stemming techniques for the Arabic language, most of those attempts still suffer from many problems such as dealing with irregular words, broken plurals words and the blind removing of affixes that lead to change in meaning of words and reducing the performance of the stemmer.

The next section will discuss a new hybrid stemming algorithm that solves the above mentioned problems. Then the algorithm will be tested against the most effective root stemmer "Khoja" and the most effective light stemmer "Light10". Those algorithms will be also included into the Arabic IR tool which will be described later in this chapter.

4.1 Proposed hybrid stemmer

In this research, the researcher proposes a new hybrid stemming algorithm – referred to as "the proposed stemmer" - that integrates between the affixes removal and lookup approaches. The proposed stemmer improves the performance of information retrieval by defining a set of morphological rules that solves many of the ambiguity problems of light stemming like broken plurals and blind removal of the affixes.

The researcher developed an Arabic morphological engine; which takes a set of patterns, affixes, and corpora as input and extracts morphological rules. These rules will be applied into words and then the stem word will be extracted depending on techniques discussed below.

The algorithm will be divided into two sections; section 4.1.1 will describe the main idea of how to extract rules, while section 4.1.2 will analyze the extracted rules and then select the best set of rules to be used in the stemmer.

4.1.1 Building Rules – Training

The main goal of this step is to define a set of rules that will be used in the stemming algorithm. In this step, Arabic morphological rules are built depending on three inputs which are:

1. Set of Affixes listed in Table 4.1, the affixes include only prefixes and suffixes, as all set of antefixes and postfixes are combined with prefixes and suffixes respectively.

Table 4.1: Affixes list

Affixes		
Prefixes	P1	ل – ب – ف – س – و – ي – ت – ن – ا
	P2	ال – لل – فل – ول – وب
	P3	ولل – وال - كال – بال – فال
Suffixes	S1	ة – ه – ي – ك – ت – ا – ن
	S2	ون – ات – ان – ين – تن – كم – هن – نا – يا – ها – تم – كن – ني – وا – ما - هم
	S3	تمل – همل – تان – تين – كمل

2. Predefined lists of patterns which are gathered from available patterns in Khoja [6] and additional patterns from Albawab [19]. Lists are shown in Table 4.2 depending on length.

Table 4.2: Arabic patterns

Length	Patterns
L4	فاعل – افعل – فعّل – تفعل – فعال – مفعل – فعول – فعيل - فعلى – فعلة
L5	افتعل – انفعل – تفاعل – تفعيل – افعال – مفاعل – متفعل – منفعل – مفتعل – مفعول – مفعال – فعلان – فعلاء – فواعل – افاعل – يفتعل – تفتعل – فاعول – فعائل – تفعلل – افعلل – مفعلل – فعالل
L6	استفعل – انفعال – افتعال – افعلال – متفاعل – مستفعل – مفاعيل – يستفعل – افعوعل – متفعلل
L7	استفعال

3. Arabic Corpus - Open Source Arabic Corpus OSAC [20] - that will be described in the next chapter.

The main idea of "building rules" step is that the word will be firstly matched against the list of predefined patterns, if there is no pattern match then we will start removing affixes and retry to match it with the predefined patterns after each removal. Figure 4.1 shows the general diagram of the proposed stemmer:

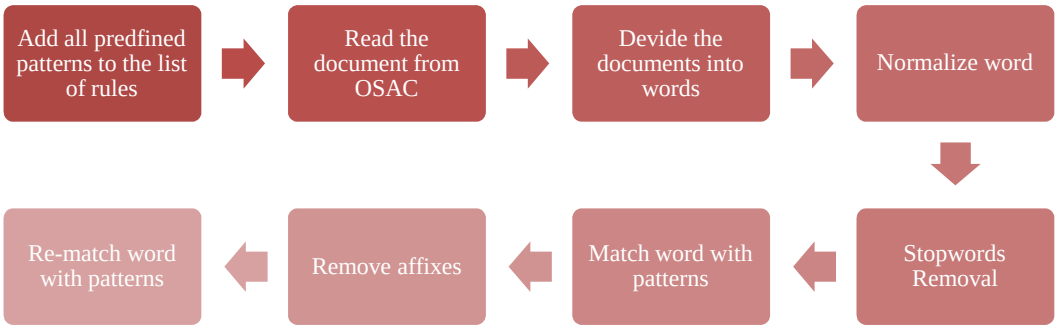


Figure 4.1: Basic steps of building rules process

∨ Flow chart of building rules process - The training phase

1. Matching the word against the Arabic pattern list before removing any affixes, the goal of this step is to solve the problem of blind affixes removal as if there is a word that starts or ends with possible prefix or suffix and the word matches one pattern before removing the affixes, then it is a valid word and the affixes in the word is a part of the original word and must be kept.

   For example the word "الوان"starts with a possible prefix "ال", but as we see the "ال" is a part of the original word and removing it will lead to have the root "وان"which has no meaning. When applying the match first then the word will be matched with the pattern of the length five "افعال"and we will return it without change.

   If the match occurs then there is no need to add any additional rule to the list of rules as all predefined patterns have been added before.

2. If the word does not match any of the predefined patterns, then we need to truncate its prefixes and suffixes to find a new rule, we will start by removing prefixes and suffixes of length three and two respectively.

   The removal process must be done depending on some constrains, firstly we start by checking the word length, if it is greater than or equal six then we will remove a prefix of length three, if not then we will check if the length is equal to five and if yes we will remove the prefix of length two.

   The same constrains will be checked to remove suffixes of length three and two. The reason of removing prefixes before suffixes will be discussed in a special section later in this chapter.

Let us take an example of this step, suppose we have a word like "المنظمات", the length of the word is eight which is greater than six but nothing from the prefixes of length three matches the first three characters "الم", so check the two characters "ال"against the set of prefixes of length two, the prefix will be found and it will be added to a new special prefix list which was established to be used only in the building rules phase and the list will now contain only "ال". The same will be done with suffixes and the special suffix list will be initialized with suffix "ات", and the remaining word will be "منظم"

3. If the remaining word length equal to three then we will stop the process and add the rule to the list of rules; the rule will consist of the special prefix list plus "فعل"plus the special suffix list.

4. If the remaining word length is equal to four then match the word against the list of Arabic patterns of length four, if the match occurs, then add a new rule, if not then try to remove one prefix or suffix according to predefined prefixes and suffixes of length one list and then add a new rule.

   When the word does not match any pattern and also there is no one prefix or suffix matched then the word will be neglected and considered as irregular word.

   By looking into previous example, the word "منظم"will be matched against the set of predefined Arabic patterns of length four, and return the pattern "مفعل, so a new rule will be added to the list of rules.

   Table 4.3 describes the matching process while Figure 4.2 shows the structure of the new rule.

Table 4.3: Matched pattern for word "منظم"

الكلمة	م	ن	ظ	م
النمط المقابل	م	ف	ع	ل

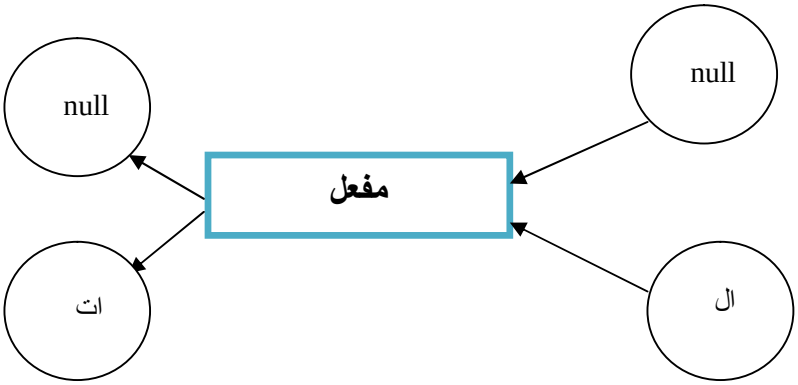


Figure 4.2: Rules tree for pattern "مفعل"

5. Match the word against the list of Arabic patterns of the same remaining length, if the match occur, then add a new rule, if not; try to remove one prefix or suffix according to predefined prefixes and suffixes of length one list, if one of prefixes or suffixes has been removed then reprocess step five with the new word length. After applying the above mentioned seven steps we will have a list of rules that will be used later in the proposed stemmer. Figure 4.3 shows the flowchart of the algorithim while Table 4.4 shows the rules of the pattern "مفعل" that result from applying algorithm in OSAC corpus.

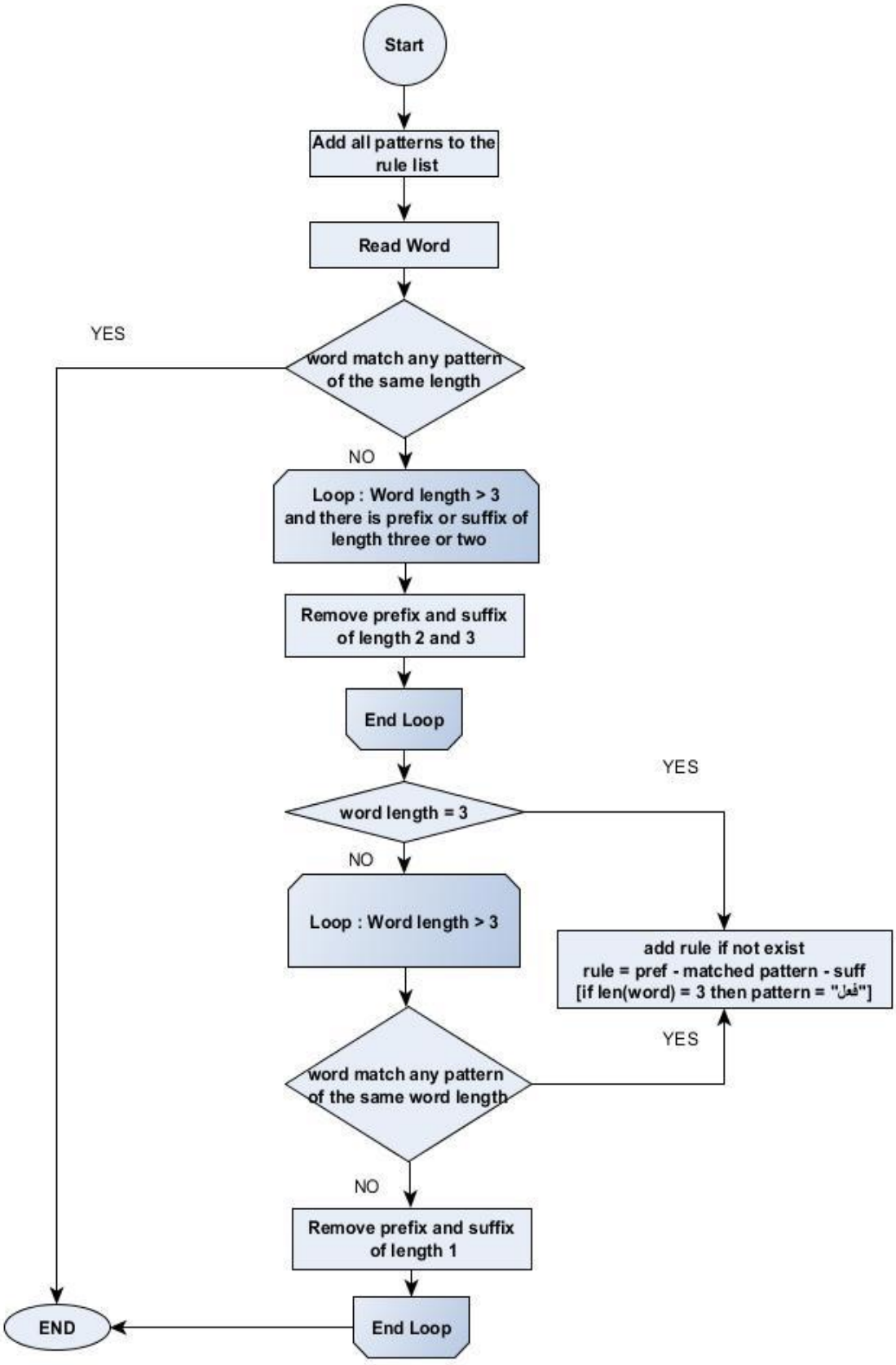


Figure 4.3: Flow chart of building rules process

Table 4.4:The rules of the pattern "مفعل"

example	rule	Suff.	pattern	Pre.	#	example	rule	Suff.	pattern	Pre.	#
و المعلمات	والمفعلات	ات	مفعل	وال	45	المنظمات	المفعلات	ات	مفعل	ال	1
مجلداته	مفعلاته	اته	مفعل		46	بمعظم	بمفعل		مفعل	ب	2
بمعزلة	بمفعلة	ة	مفعل	ب	47	المدمر - الموقع	المفعل		مفعل	ال	3
ومؤكدا - ومركزا	ومفعلا	ا	مفعل	و	48	موظفيها	مفعليها	يها	مفعل		4
متخطيا - متمنيا	مفعليا	يا	مفعل		49	موقف	مفعل		مفعل		5
ومدججة - و مميزة - وممثلة	ومفعلة	ة	مفعل	و	50	ومؤلم	ومفعل		مفعل	و	6
وتمركز	وتمفعل		مفعل	وت	51	معدلات	مفعلات	ات	مفعل		7
وبمعزل	وبمفعل		مفعل	وب	52	لمرحلة - لمحطة	لمفعلة	ة	مفعل	ل	8
بالمجمعات	بالمفعلات	ات	مفعل	بال	53	لمنصب	لمفعل		مفعل	ل	9
وممثلي	ومفعلي	ي	مفعل	و	54	ومؤسسته - ومصلحته	ومفعلته	ته	مفعل	و	10
لمخصصاتها	لمفعلاتها	اتها	مفعل	ل	55	مرحبا	مفعلا	ا	مفعل		11
لموقعها	لمفعلها	ها	مفعل	ل	56	لمؤسسته	لمفعلته	ته	مفعل	ل	12
المؤهلتين	المفعلتين	تين	مفعل	ال	57	الموظفون	المفعلون	ون	مفعل	ال	13
المعنيتان	المفعلتان	تان	مفعل	ال	58	منصبها	مفعلها	ها	مفعل		14
المسلحان	المفعلان	ان	مفعل	ال	59	معرفته	مفعلته	ته	مفعل		15
الممثلين - المسلحين	المفعليين	يين	مفعل	ال	60	المشرعين	المفعلين	ين	مفعل	ال	16
بمركبات	بمفعلات	ات	مفعل	ب	61	للمنصب	للمفعل		مفعل	لل	17
لممثلي - لمطربي	لمفعلي	ي	مفعل	ل	62	بالموعد	بالمفعل		مفعل	بال	18
مؤيديه - موظفيه	مفعليه	يه	مفعل		63	بمقتله	بمفعله	ه	مفعل	ب	19
مبيعاتنا - مدرجاتنا	مفعلاتنا	اتنا	مفعل		64	موعدنا	مفعلنا	نا	مفعل		20
مخصصاتها	مفعلاتها	اتها	مفعل		65	مقربون	مفعلون	ون	مفعل		21
مصلحتنا	مفعلتنا	تنا	مفعل		66	مصرعهم	مفعلهم	هم	مفعل		22
لموظفين	لمفعلين	ين	مفعل	ل	67	مسلحين	مفعلين	ين	مفعل		23
لمؤسسات	لمفعلات	ات	مفعل	ل	68	مدرستي	مفعلتي	تي	مفعل		24
وميولهم - ومدربهم	ومفعلهم	هم	مفعل	و	69	المدفعية	المفعلية	ية	مفعل	ال	25
ومسلحون - ومشرعون	ومفعلون	ون	مفعل	و	70	ومنظمات	ومفعلات	ات	مفعل	و	26
للمركزي - للمحمدي	للمفعلي	ي	مفعل	لل	71	مغربية	مفعلية	ية	مفعل		27
معنويات - مجريات	مفعليات	يات	مفعل		72	معرفتها	مفعلتها	تها	مفعل		28
بمجندين - بمصدرين	بمفعلين	ين	مفعل	ب	73	مصنعه	مفعله	ه	مفعل		29
مصرفيين	مفعليين	يين	مفعل		74	مؤجلان - موقعان	مفعلان	ان	مفعل		30
بمنتجاتها	بمفعلاتها	اتها	مفعل	ب	75	لمدربه	لمفعله	ه	مفعل	ل	31
مخططاتهم	مفعلاتهم	اتهم	مفعل		76	والمصدر	والمفعل		مفعل	وال	32

example	rule	Suff.	pattern	Pre.	#	example	rule	Suff.	pattern	Pre.	#
مؤجلتان - مزودتان	مفعلتان	تان	مفعل		77	والمسلمين	والمفعلين	ين	مفعل	وال	33
فموقع - فمجرد	فمفعل		مفعل	ف	78	بالمسلمين	بالمفعلين	ين	مفعل	بال	34
ومنتجاتها	ومفعلاتها	اتها	مفعل	و	79	مقدمتهم	مفعلتهم	تهم	مفعل		35
ومفكرين	ومفعلين	ين	مفعل	و	80	مشرعنة	مفعلنة	نة	مفعل		36
مؤجلتين	مفعلتين	تين	مفعل		81	بموعدها	بمفعلها	ها	مفعل	ب	37
المؤسسه	المفعله	ه	مفعل	ال	82	بتمركز	بتمفعل		مفعل	بت	38
مظهرك - موقعك	مفعلك	ك	مفعل		83	تمركزا	تمفعلا	ا	مفعل	ت	39
مظهرهما - مصرعهما	مفعلهما	هما	مفعل		84	للمبررات	للمفعلات	ات	مفعل	لل	40
ومصدره - وموقفه	ومفعله	ه	مفعل	و	85	ومغربية	ومفعلية	ية	مفعل	و	41
والمصابون	والمفعلون	ون	مفعل	وال	86	للمعلمين	للمفعلين	ين	مفعل	لل	42
مخططهن	مفعلهن	هن	مفعل		87	فالمصدر - فالمخرج	فالمفعل		مفعل	فال	43
						مجلداته	مفعلاته	اته	مفعل		44

The result of the training phase was a set of rules consisting of 4398 rules generated from 1852631 unique words spread over forty two patterns. Table 4.5 shows the distribution of these rules according to patterns.

Table 4.5: Distribution of rules over patterns

#	الوزن	عدد القواعد - Rules	#	الوزن	عدد القواعد - Rules	#	الوزن	عدد القواعد - Rules
1	فعال	571	15	فعائل	75	29	فعالل	25
2	فعيل	542	16	مفتعل	75	30	فعالي	24
3	فاعل	492	17	فواعل	70	31	متفعلل	23
4	فعول	427	18	مفعيل	61	32	تفاعيل	22
5	افعل	361	19	افتعال	57	33	مفعلل	18
6	تفعل	272	20	استفعل	54	34	افعوعل	12
7	تفاعل	127	21	فعلان	54	35	فعلة	12
8	افعال	122	22	تفعيل	52	36	فاعلة	11
9	مفاعل	116	23	تفتعل	51	37	فعالة	9
10	فاعول	93	24	مفعول	49	38	مفعلة	8
11	مفعل	87	25	منفعل	43	39	تفعلة	6
12	افاعل	85	26	مستفعل	42	40	مفاعلة	6
13	افتعل	81	27	يفتعل	41	41	فعولة	5
14	انفعل	78	28	فعلى	26	42	افعلة	4

v The morphological Structure of Arabic words

The technique used in the previous section depends on the morphological structure of the Arabic word, so we need to describe and analyze the structure to determine the

reason of removing the prefixes before suffixes and to reorder the predefined patterns. The morphological structure of the Arabic word described in Figure 4.4.

Figure 4.4: morphological Structure of Arabic word

Figure 4.4 describes the structure of the word; firstly add infixes to the root to generate the stem form and then attach the prefixes and suffixes to generate the full word.

A study has been done by the researcher to show the average occurrences of suffixes and prefixes into Arabic words; the study include analyze of more than 46,000 words selected randomly from the OSAC corpus. Table 4.6 shows the distribution of suffix and prefix into these words.

Table 4.6: Distribution of prefixes and suffixes into Arabic words

	Number of words	Percent
Only prefixes	15166	32.13%
Only suffixes	12022	25.48%
Has prefixes and suffixes	10169	21.54%
None	9838	20.85%
Total	47195	100%

Table 4.6 shows that more than 20% of the Arabic words do not have any prefixes or suffixes, so blind removal of the affixes will affect those words and will remove original letters from the word. Also it is noticeable from the table that about 33% of the words will have prefixes only and the percent is greater than the percent of words that have suffixes only by about 7%. The researcher also do another study which showed the distribution of the number of words according to the number of prefixes and suffixes, as exhibited in Figure 4.5.

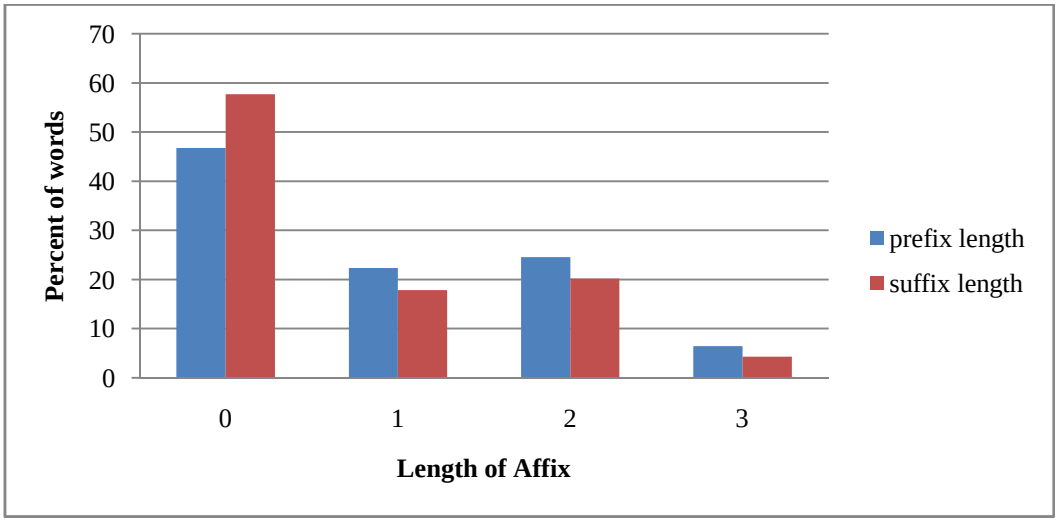


Figure 4.5: Distribution of the number of words according to the number of prefixes and suffixes

From Figure 4.5 it can be confirmed that blind remove of affixes will affect the stemming process, as there is more than 45% of words do not have prefixes and about 57% do not have suffixes. Also the percent of words with prefixes is always greater than words with suffixes for all lengths, and as mentioned before, the percent of words with prefixes only is greater than words with suffixes only by about 7%, the researcher decided to remove prefixes before suffixes in the proposed algorithm as it has more popular occurrences.

A study of the frequency of all patterns was conducted; its aim of was to reorder the patterns of the same length according to the number of occurrences. Ordering those patterns will lead to better performance as the word that matches more than one pattern will be matched to the most popular one. Figures [4.6, 4.7, 4.8] show the distribution of words in patterns of the length five, four and six respectively.

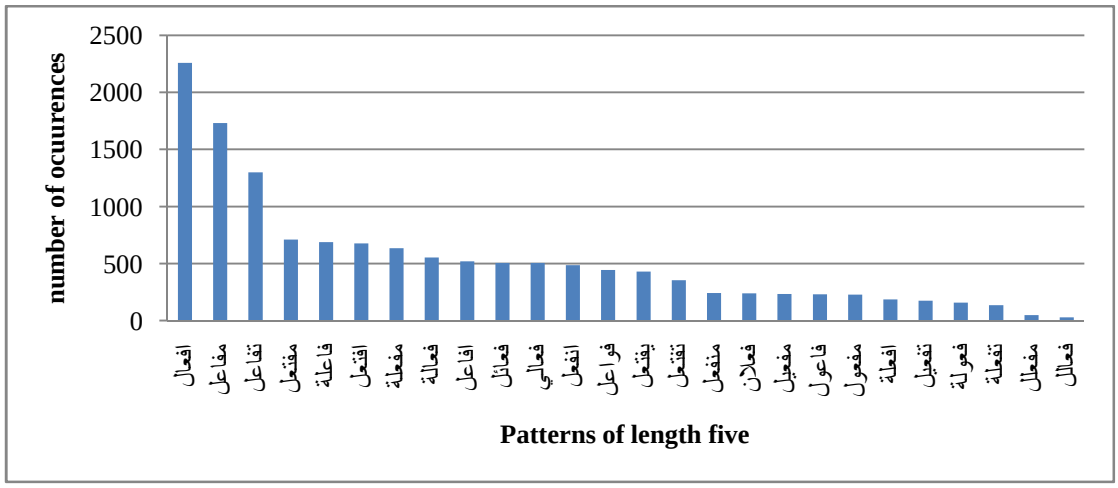


Figure 4.6: The distribution of words in patterns of the length five

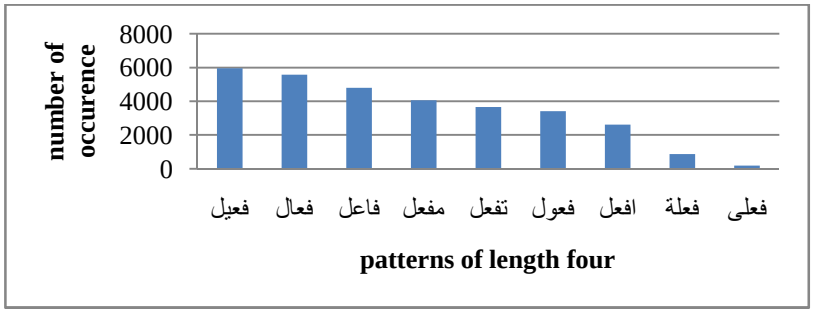


Figure 4.7: The distribution of words in patterns of the length four

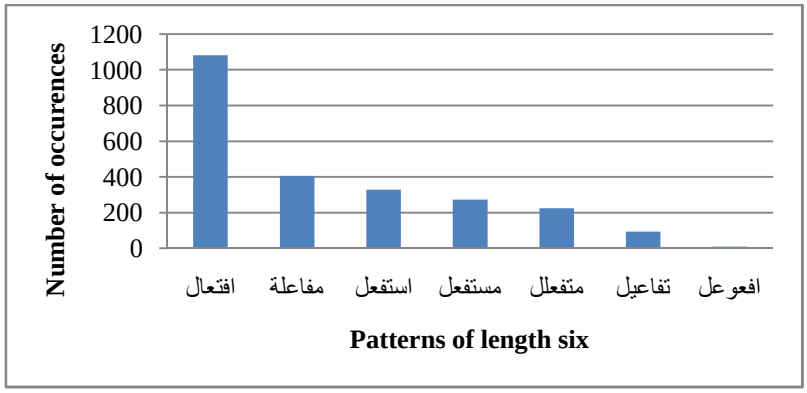


Figure 4.8: The distribution of words in patterns of the length six

Figure 4.6 shows the distribution of words into patterns of length five, the figure shows that the pattern "افعال" is the most popular pattern while the pattern "فعالل"has the least occurrences. The same for patterns with length four as shown in Figure 4.7; "فعيل" is the most popular one for this length while "فعلى" is the least one. For patterns with length six the pattern "افتعال" has the most number of occurrences while "افعوعل"has the lowest number as shown in Figure 4.8.

Table 4.7 shows the ordered list of the Arabic patterns, depending on the study above.

Table 4.7: The ordered list of the Arabic patterns

Length	Patterns
L4	فعيل – فعال – فاعل – مفعل – تفعل – فعول –افعل –فعلة - فعلى
L5	افعال – مفاعل – تفاعل – مفتعل – فاعلة – افتعل – مفعلة – فعالة – افاعل – فعائل – فعالي – انفعل – فواعل – يفتعل – تفتعل – منفعل – فاعول – مفعول – افعلة – تفعيل – فعولة – تفعلة – مفعلل - فعالل
L6	افتعال – مفاعلة – استفعل – مستفعل – متفعلل – تفاعيل – افعوعل – افعلال - مفاعيل
L7	استفعال

4.1.2 Rule-based Stemmer

The proposed stemmer has been developed depending on the rule list that has been derived in the previous section. The rule list will consist of all possible combinations of prefixes and suffixes that can be added to the predefined list of Arabic patterns from Table 4.7 to create a new form of the patterns. Figure 4.2 is an example of the generated rules for the pattern "مفعل".

For a new input word, the word is string matched to the list of rules, matching was done only with the rules that have the same length of the word, and by matching prefixes and suffixes the pattern will be extracted.

The list of rules that has been generated needs to be reviewed before use, as any rule that appears only from one word, it will be considered as irregular word and needs to be removed from the list. Table 4.8 shows an example of selected irregular words depending on previous conditions.

Table 4.8: Sample of extracted irregular words from rules list

#	الكلمة	The rule		
		prefix	pattern	Suffix
1	بريطانيا		فعيل	انيا
2	نيويورك	ني	فعول	ك
3	الامريكيين	ال	افعل	كيين
4	لواشنطن	لو	افعل	ن
5	السيناتور	السين	فعول	
6	اوبينيون		افعل	نيون
7	بوليسية	ب	فعيل	ية
8	الباكستاني	ال	فاعل	تاني
9	التلفزيون	ال	تفعل	يون
10	النارمع	الن	افعل	
11	وبلغاريا	وب	فعال	يا
12	لطالبان	ل	فاعل	ان
13	افغانستان	اف	فاعل	تان
14	البريطانيون	ال	فعيل	انيون
15	بلاكويل	بلا	فعيل	
16	تلغراف	تل	فعال	
17	بفيتنام	بفي	فعال	
18	الانسولين	ال	انفعل	ين
19	الدواءكما	ال	فواعل	ما
20	اللوكيميا	الل	فعيل	يا
21	اتلانتا		افعال	تا

#	الكلمة	The rule		
		prefix	pattern	suffix
31	لليونيسيف	لليون	فعيل	
32	فلوريدا	فل	فعيل	ا
33	الفياغرا	الف	فاعل	ا
34	الفارادو	الف	افاعل	
35	البروستات	ال	فعول	تات
36	لواينر	لو	افعل	
37	باترسون	ب	افعل	ون
38	اوغوستا		فاعول	تا
39	اللمفاوي	الل	مفعل	ي
40	موريتانيا		مفعيل	انيا
41	البيولوجي	البي	فعول	ي
42	ستاركوف	ست	فاعول	
43	فيروزن	ف	فعول	ن
44	ترانزيت		تفاعل	يت
45	يونايتد	يون	افعل	
46	ستراوسكان	س	تفاعل	كان
47	فودافون	ف	فعال	ون
48	ميكانيك		فعلان	يك
49	فنزويلا	فن	فعيل	ا
50	يوتيوب	يو	فعول	
51	فيروسات	ف	فعول	ات

The rule			الكلمة	#	The rule			الكلمة	#
suffix	pattern	prefix			Suffix	pattern	prefix		
	فاعل	لس	لسكايب	52	ين	استفعل	ال	الاستروجين	22
ن	فعيل	اس	اسوسيشن	53	ون	فاعول	و	والقولون	23
	فواعل	ا	ادوارد	54		فعول	نيت	نيتينول	24
	فعيل	السين	السيناريو	55	ين	فعول	الني	النيتروجين	25
يون	فاعل	ل	لهامليون	56	ني	فعيل	تل	تلفزيوني	26
ا	فعول	كالفو	كالفورمولا	57	نيا	فعول	كال	كاليفورنيا	27
وا	افعل	وال	والاونروا	58		فعول	تلي	تليسكوب	28
ن	فاعل	س	ستايشن	59	يا	فعول	بي	بيولوجيا	29
ين	فعيل	ا	ايثيلين	60	ون	فعال	الب	البنتاغون	30

From Table 4.8 we can notice that we do not need to define a list of irregular words, as any word that does not match rule from the remaining rules will be considered as irregular and will be returned without stemming. This process will also manage the problem of missing space between words, for example the word number 19 in the previous table which shows the word "الدواءكما", which is originally the combination of two words "الدواء" and "كما", so it was considered as an irregular word because there was no space between the two words.

v The proposed stemmer Algorithm

To start describing the proposed stemmer steps, first we define a set of diacritical marks, punctuations and a list of stopwords as shown in Table 4.9.

Table 4.9: A set of diacritical marks, punctuations and a list of stopwords

Diacritical marks	ً- ٌ- ٍ- َ- ُ- ِ- ّ- ْ									
Punctuations	.	÷	~	.	;	{				
	,	×	=	+	/	}				
	?		؟	'	%	\	)			
	:	>		!	^	?	(			
	"	<		@	&	$	_			
	'	|		#	*	'	-			
Stopwords	بعد	كذلك	فقط	ومن	هؤلاء	هو	وقد	اضحى	او	ضمن	فكان
	ضد	التي	ثم	لا	فإن	عنها	كانت	ظل	و	اول	ستكون
	يلي	وبين	هذه	ليسب	فيه	منه	لذلك	مابرح	ما	وله	مما
	الى	فيها	أنه	وكانت	ذلك	بها	أمام	مافتئ	لا	ذات	أبو
	في	عليها	تكون	أي	لو	وفي	هناك	ماانفك	الي	اي	بإن

	من	إن	قد	ما	عند	فهو	قبل	بات	إلي	بدلا	الذي
	حتى	وعلى	بين	عنه	اللذين	تحت	معه	صار	مازال	اليها	اليه
	وهو	لكن	جدا	حول	كل	لها	يوم	ليس	لازال	انه	يمكن
	يكون	عن	لن	دون	بد	أو	منها	إن	لايزال	الذين	بهذا
	به	مساء	نحو	مع	لدى	إذ	إلى	كأن	مايزال	فانه	لدي
	وليس	ليس	كان	لكنه	وثي	علي	إذا	ليت	اصبح	وان	وأن
	أحد	منذ	لهم	ولكن	أن	عليه	هل	لعل	أصبح	والذي	وهي
	على	الذي	لأن	له	ومع	كما	حيث	لاسيما	أمسى	وهذا	وأبو
	وكان	أما	اليوم	هذا	فقد	كيف	هي	ولايزال	امسى	لهذا	آل

The stemming process will proceeds by the following steps:

§ Tokenization: This is a necessary and meaningful step in natural language processing. The function of a tokenizer is to break down a text stream into segments so that they can be introduced into a morphological sensor or a position tagger. The tokenizer is responsible for defining boundaries of a word; it is based mainly on the white spaces and punctuation marks as delimiters between words or major segments.

In our algorithm we will use the separator " \r\n\t.,;:'\"()?!" which depend on new lines, white spaces, tabs and some punctuation marks.

§ Normalization: In the proposed algorithm, normalization will include the following steps:
- Removing the shadda and the diacritics – Table 4.9.
- Removing punctuations – Table4.9.
- Removing all numbers.
- Replacing “أ”, “إ” and “آ” by “ا”
- Replacing “ي” by “ى” at the end of the words.
- Replacing the sequence “يء” by “ئ”.
- Removing the tatweel character “-”.
- Remove stopwords – Table 4.9.

After preprocessing steps, each word is matched against a list of predefined rules which has been generated previously. If the word matched a rule then remove all affixes depending on that rule and take the remaining word as stem word, if not then return the word itself.

Returning the stemmed words is not enough; as some words may match broken plural pattern, and then it needs to be converted to the singular form of that pattern. So our

predefined patterns need to be classified to two parts: broken plurals patterns and normal patterns.

The broken plurals patterns are important for Arabic text as it form around 10% of any Arabic content; the word takes different morphological form than the singular one so it needs to be reformed to the singular form. The researcher defines a dictionary of these patterns and their singular forms, and after getting the stemmed word from the previous step, then it will be matched against this dictionary, if the pattern is one of broken plurals patterns then the word is converted to its singular form, and if not it will be returned without change. Table 4.10 describes the broken plurals patterns and all singular forms of it.

Table 4.10: Broken plurals and its singular form(s)

Broken Pattern	Singular Form	Plural Example	Singular Form
مفاعيل	مفعول	مجانين	مجنون
افعال	فعل	اصوات	صوت
فعلاء	فعل – فعال – فاعل - فعيل	سمحاء – جبناء – عقلاء - اطباء	سمح – جبان – عاقل – طبيب
فواعل	فاعل - فوعل	شوارع - مواسم	شارع – موسم
فعائل	فعيل	ضمائر	ضمير
فعايا	فعيه	خلايا	خليه
فعاليل	فعويل - فعليل	ملايين - براميل	مليون – برميل
افعياء	فعي	اغبياء	غبي
فواعيل	فاعول	طوابير	طابور

From the table above we can notice that some broken plurals have more than one singular form, so when the word is matched against one of them, it is needed to return all singular forms of that pattern.

4.2 New Arabic IR tool kit

The researcher developed a new Arabic toolkit for information retrieval purposes, the toolkit allows user to define lists of constants used for preprocessing like punctuations, prefixes, suffixes and stopwords. The user can also choose between stemming algorithms or add new one “for developers” as the tool will be open source project. The tool also provides the ability for user to mix between many normalization techniques, choose the weighting technique and define the tokenizer.

The tool provides the ability of testing any stemmer by applying one of the built in text classification techniques or adding a new one, the results of classification processes can be compared according to many measures and can be viewed as charts.

The toolkit has been developed using java programming language with JDK 1.6. It was divided to four main screens which are constants, processing, classification and visualization.

Figure 4.9 shows the main functions of the new toolkit starting from defining constants to the process of comparing results; each process will be described separately.

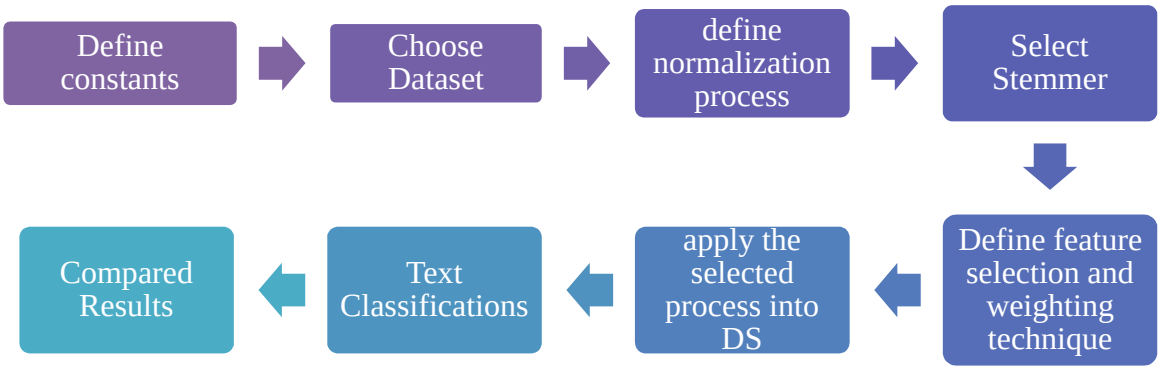


Figure 4.9: Main functions of new Arabic IR toolkit

Figure 4.10 shows the constant screen, from this screen user can define the lists of stopwords, punctuations, prefixes, suffixes and diacritics. The default lists are the ones that were used in the proposed stemmer which has been described in the previous section. User can add new item or select new list from his device “csv format, one item per line”, and load it with the ability of keeping the original or override it. User can also delete any selected item from the list and finally save the new list as csv file.

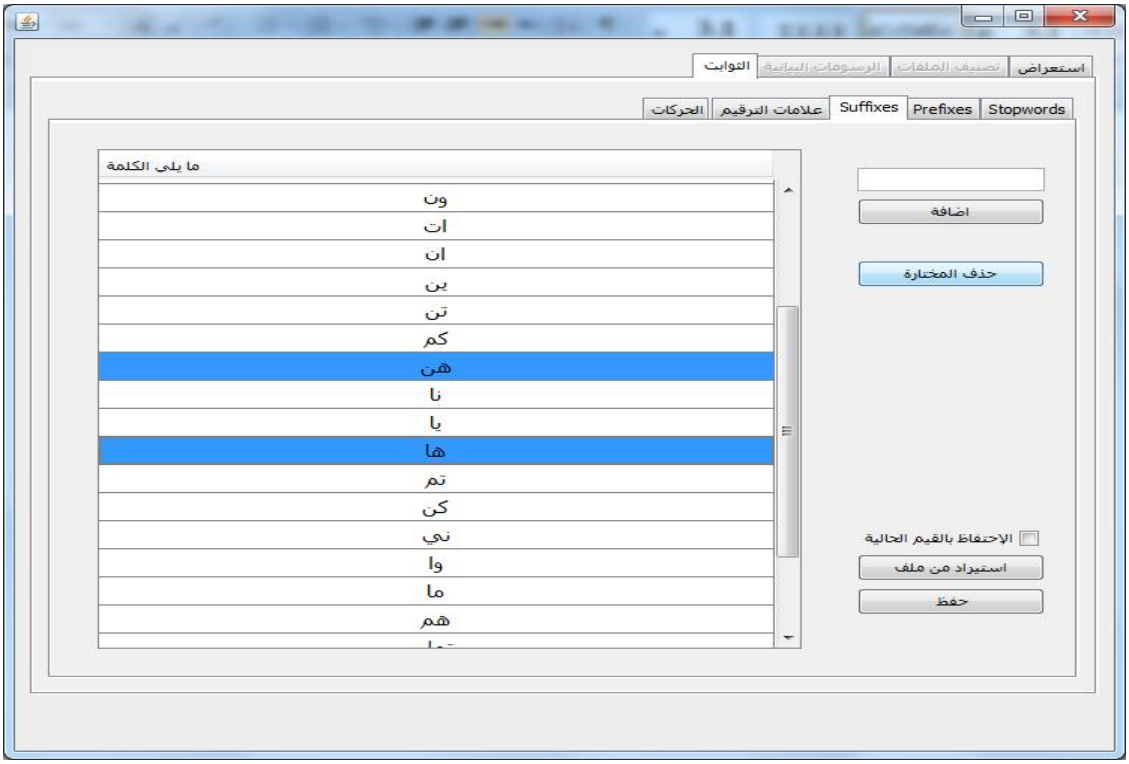


Figure 4.10: The constant screen that allows user to define a set of constancies used in the tool

For prefixes or suffixes, user does not need to specify the length of the item; once the item is added, it is directly sorted to the specific affixes list according to its length, so the developers can use these lists to build his own stemmer. There is no limitation for the item length, and the list can be expanded to include any new item.

Figure 4.11 shows that the user can select a single file "فتح ملف" or select any data set by pressing "فتح مجلد", the selected folder will represent the data set itself while the inline folders considered as classes; each class must contain text files that represents the data of this class.

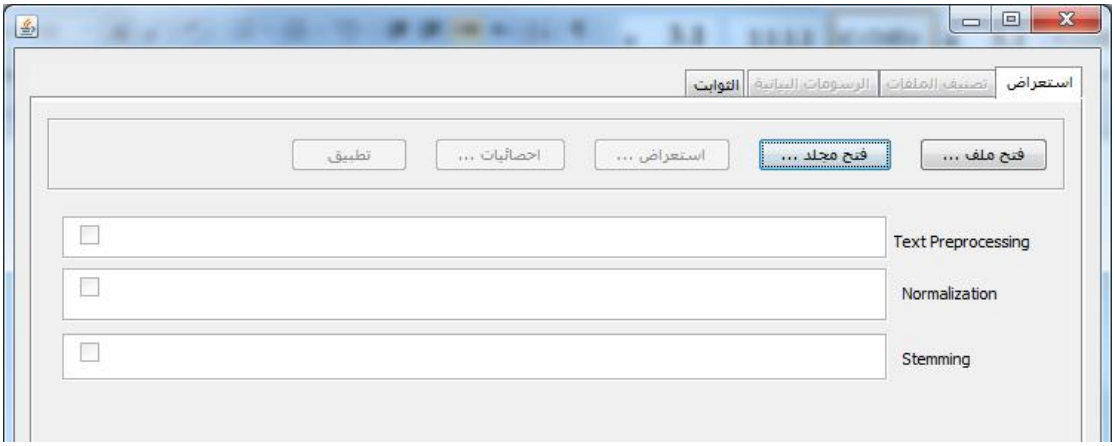


Figure 4.11: The processing screen, user can select to load one file or data set

As shown in Figure 4.11 that before loading the data set many of tool functionalities will be disabled as these functionalities depend on the data which will be loaded. The figure also shows three processes that can be applied to the loaded data set, text preprocessing, normalization and stemming. Each one will be described later, but now we need to know that the default functionality of the tool is to load data without applying any stemming algorithm, applying all sets of normalization, and tokenize data using the default tokenizer of the proposed stemmer.

Figure 4.12 shows the result which is displayed after loading the data set, the table will consist of each word, its normalized form and the stemming form. In the next tab we can find the stopwords that have been detected according to the predefined list of stopwords. We can also notice that all other functionalities have been enabled and can be viewed.

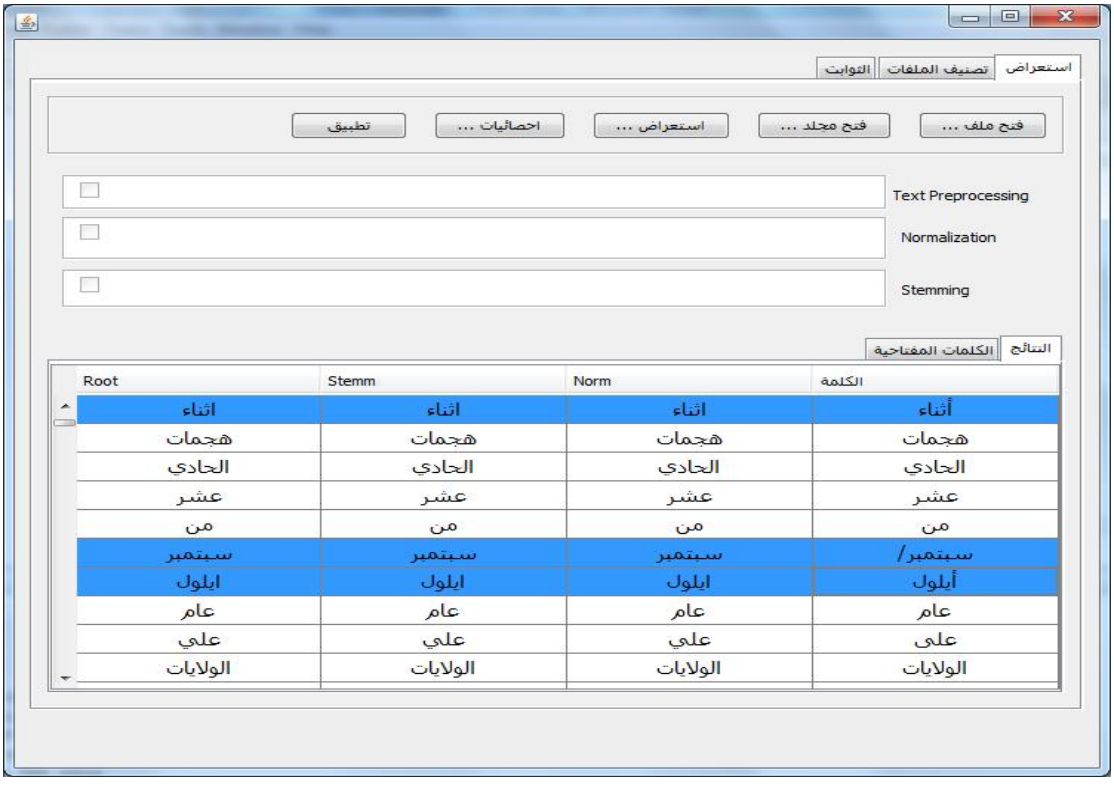


Figure 4.12: Table of results, which consist of the word, norm form and the stem

After loading the data we can then apply many processes on it with no need to reload it. The user can customize the normalization process by clicking on normalization panel; Figure 4.13 shows the screen of customizing normalization process. In this screen the user can check any one from the eight available normalization processes

and mix between them. For developers, they can add any normalization step and apply it to the normalization class.

Figure 4.13: Normalization techniques

The default for the tool is that all of these techniques are selected. The user can also select one of three built in stemming algorithms, Khoja, Light 10 and the proposed stemmer. Figure 4.14 shows the screen of selecting these stemmers which appear by clicking the stemming panel. The user can edit the default of each stemmer by applying any normalizing technique or by defining new tokenizer; a lot of combinations can be done and the effect of these combinations can be determined by comparing the results.

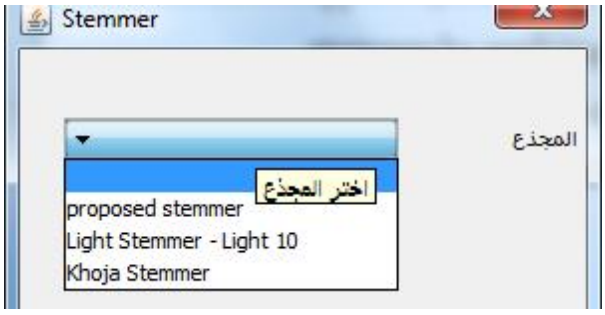


Figure 4.14: Stemmers' algorithms

Figure 4.15 shows that user can also define the tokenizer and select from different term weighting techniques; word count, term frequency, inverse document frequency and term frequency/inverse document frequency are the four feature weighting techniques the tool provides. Selecting one of these techniques depends on the text classification technique that will be used, so when using naïve base technique for text

classification then the best choice is to choose word count technique as the naïve depends on the probability of word occurrences. K-NN classifier is very sensitive to term weighting schemes because it depends on distance function to determine the nearest neighbors, so user needs to normalize data before classification. Therefore, the user should be careful when choosing these values.

The tool provides feature selection technique depending on the number of occurrences of each feature "word", user can define the number of words to keep per class " الكلمات في الفئة"these words will be selected as they are the most frequent within each class. User can also define the minimum term frequency "عدد مرات التكرار" that specifies how many times the word must appear per class to be considered as a feature.

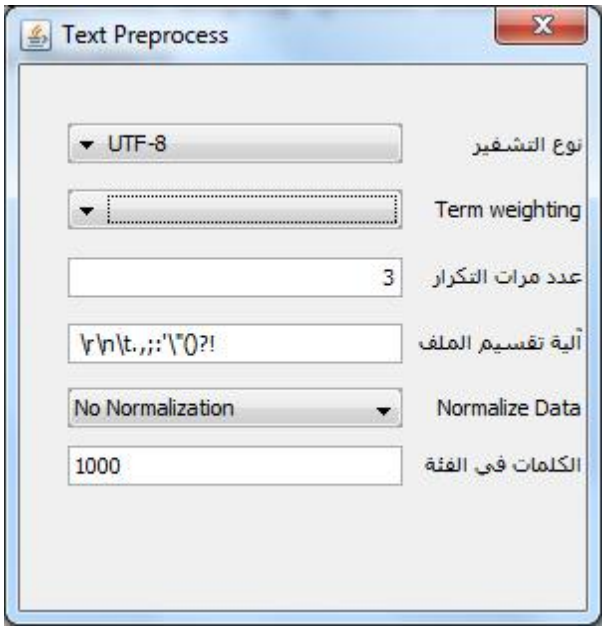
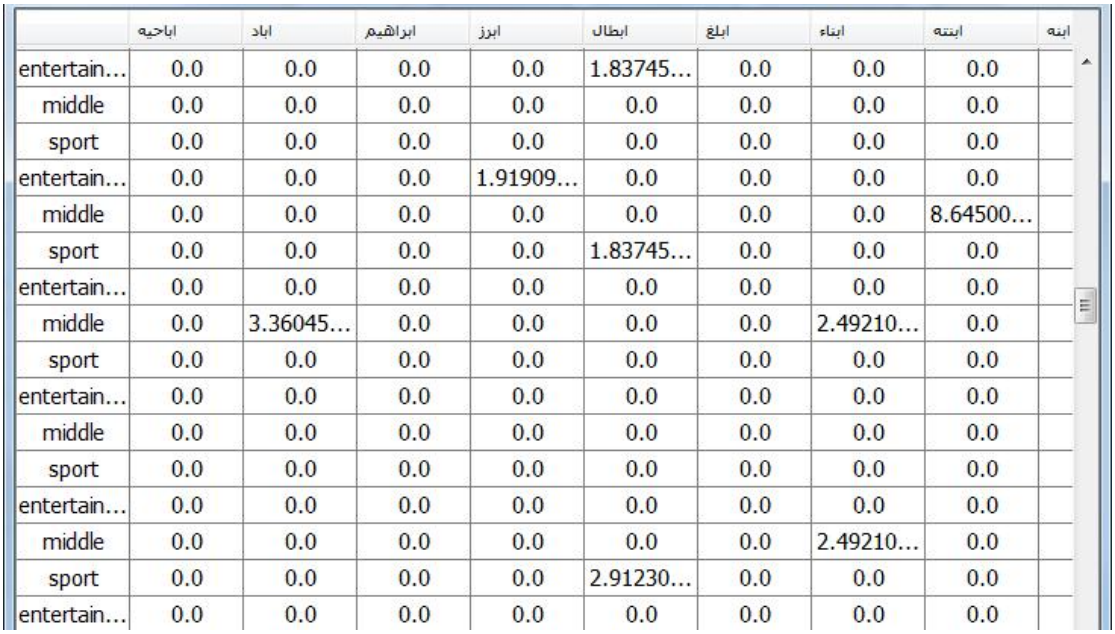


Figure 4.15: Text preprocess operations; user can specify a specific value for each field

User can change any value from the previous terms; select any normalization technique, stemmer algorithm and tokenizer; then apply these changes by clicking the "تطبيق" button with no need to reload data. Figure 4.16 shows the selected features and its weighting according to each file, these values change depending on user input for different terms. The columns represent the selected words from all classes "selected features", while rows represent all files from the corpus labeled by its class.



	اباحيه	اباد	ابراهيم	ابرز	ابطال	ابلغ	ابناء	ابنته	ابنه
entertain...	0.0	0.0	0.0	0.0	1.83745...	0.0	0.0	0.0	
middle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
sport	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
entertain...	0.0	0.0	0.0	1.91909...	0.0	0.0	0.0	0.0	
middle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	8.64500...	
sport	0.0	0.0	0.0	0.0	1.83745...	0.0	0.0	0.0	
entertain...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
middle	0.0	3.36045...	0.0	0.0	0.0	0.0	2.49210...	0.0	
sport	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
entertain...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
middle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
sport	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
entertain...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
middle	0.0	0.0	0.0	0.0	0.0	0.0	2.49210...	0.0	
sport	0.0	0.0	0.0	0.0	2.91230...	0.0	0.0	0.0	
entertain...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Figure 4.16: Features weight value according to each file

User can get statistical data - Figure 4.17 - about the executed process like number of files loaded, number of attributes selected per class, number of files per class, normalization time, stemming time and the value of each term while executing these process. User can choose from screen to display attributes per class only, number of files or to display both together.

Two text classification algorithms were included in the tool, Naïve base multinomial and K-NN as these algorithms can show the effect of term weighting and normalization techniques used because the first one depends on probability of word occurrences, while the second is affected by the distances between features. Figure 4.18 shows that user can select between the two classification algorithms and specify the percent of training data. The results of classification will show the accuracy of the classification process, measure the recall and precision for each class, and the overall average per classes. The measurement equation can be described as following:

$$precision = \frac{true\ positive}{true\ positive + false\ positive} \times 100 \qquad \textbf{4.1}$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \times 100 \qquad \textbf{4.2}$$

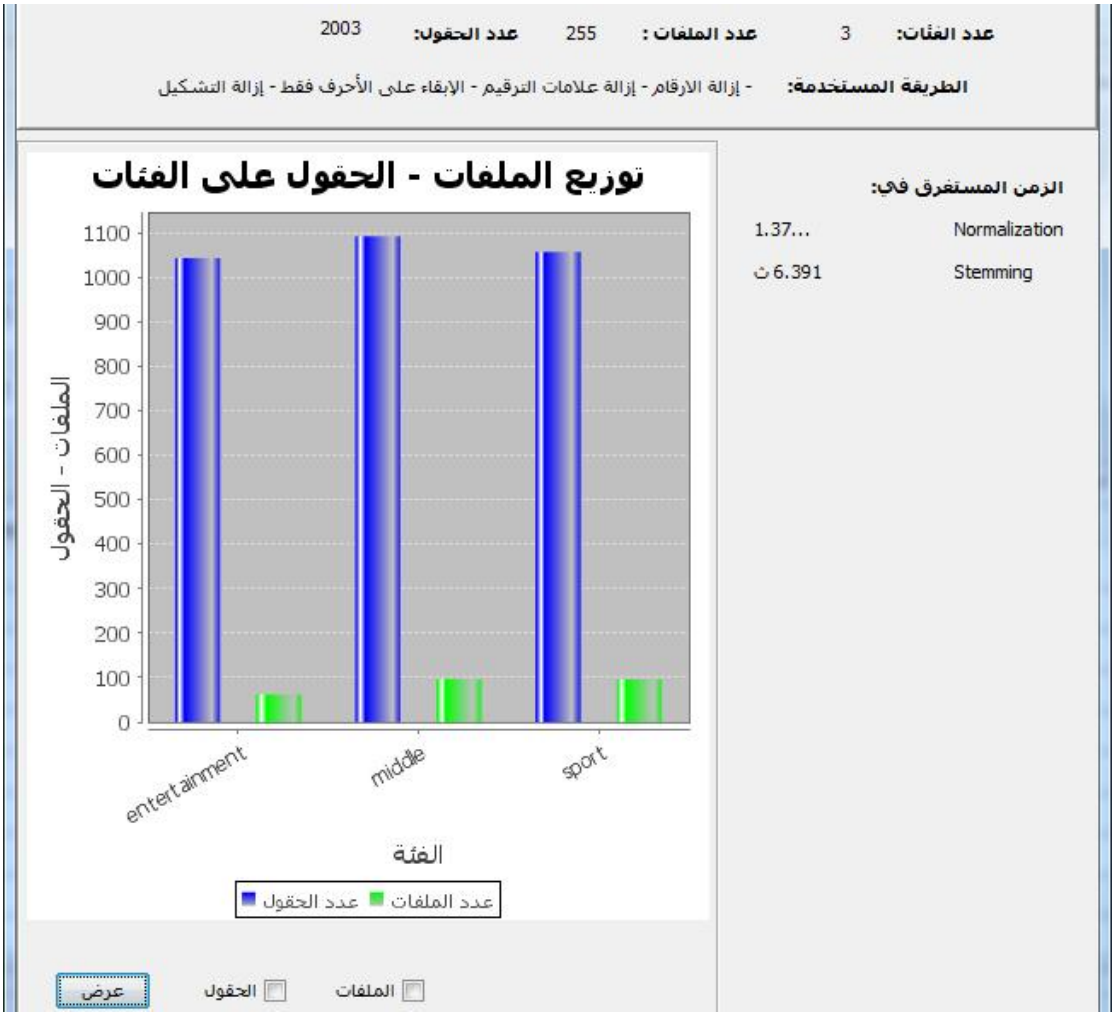


Figure 4.17: Statistical data for the stemming process

In the formulas above, for category c, the true positive is the number of documents that belongs to category c and correctly classified as category c; the false positive is the number of documents that does not belong to category c and incorrectly classified as category c; the false negative is the number of documents that belong to category c and incorrectly classified as non-category c by a classifier.

The harmonic mean of precision and recall:

$$F_1(recall, precision) = \frac{2 \times recall \times precision}{recall + precision} \qquad \textbf{4.3}$$

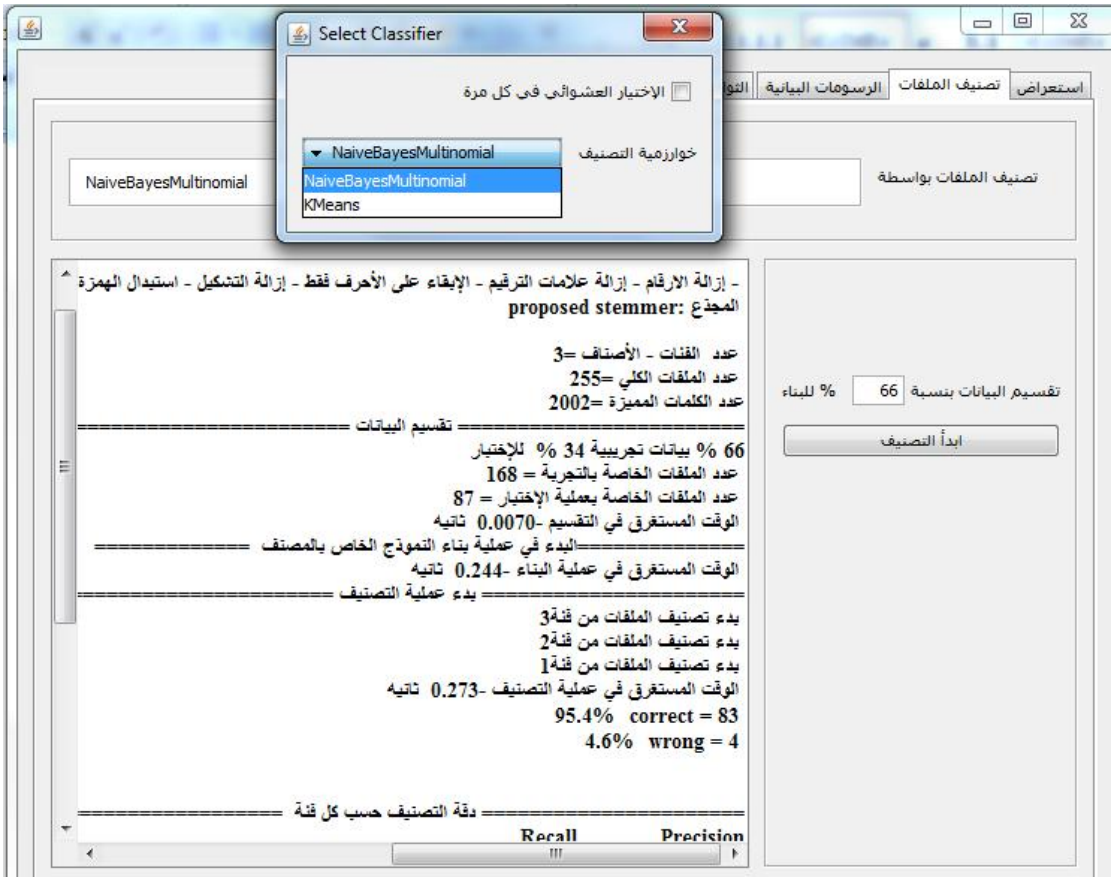


Figure 4.18: Classification screen that allows user to select between classification techniques and get results

From Figure 4.18 we can see that the classification screen provides us with full data about the classification process starting from normalization techniques used, stemmer algorithm, information about data set selected, time for split data, time for building model and classification, and finally the measurements according to each class and the overall average for all classes.

The last thing that the tool provides is the visualization for results; Figure 4.19 shows the comparisons that can be done between each process that had been done. User can make a comparison depending on number of attributes, time taken for stemming process, and accuracy of stemmer displaying the average accuracy or according to the two classification techniques separated. The list in the right side of screen represents the processes on the data. To make a comparison user must select at least two processes to compare between. Then select the comparison method from the buttons above, the diagram in Figure 4.19 shows the comparison between three stemmer algorithms according to number of attributes.

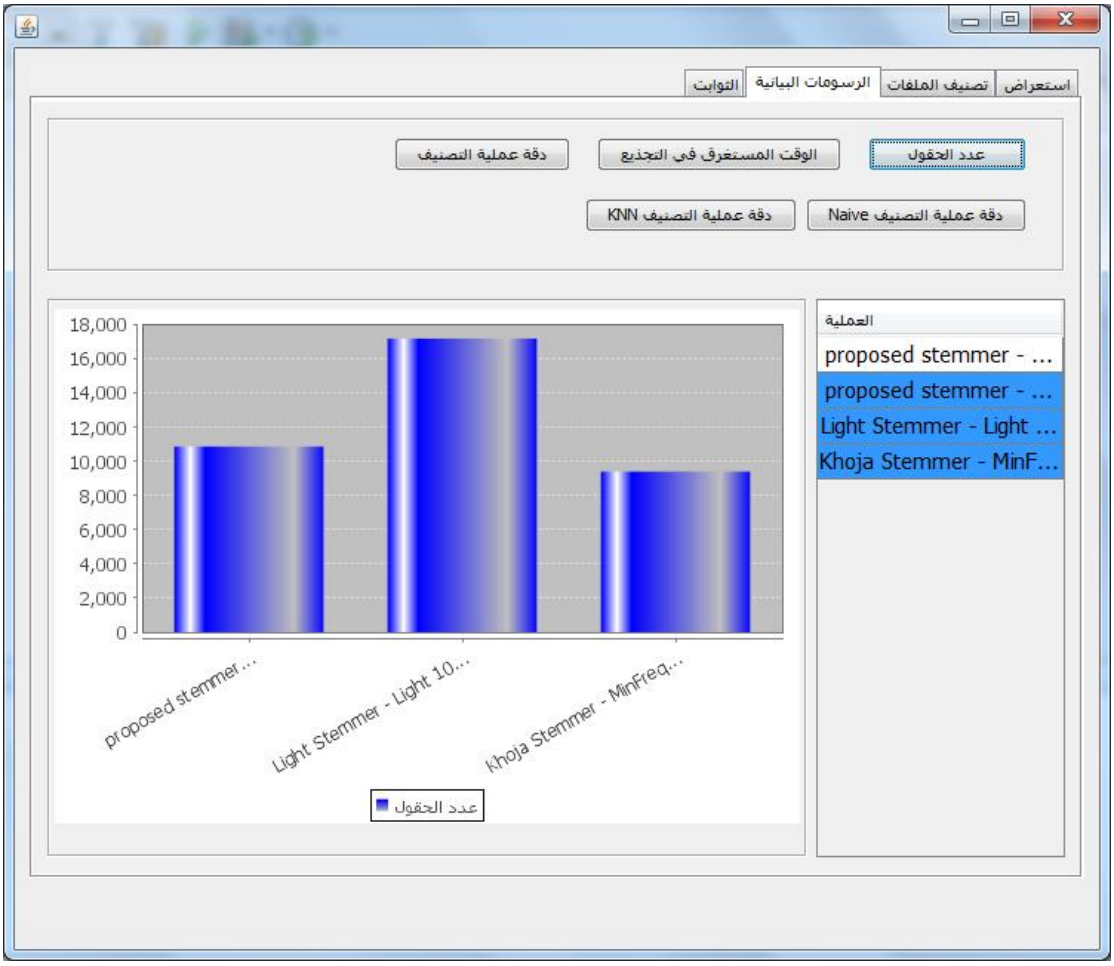


Figure 4.19: Visualization screen that allows user to do comparisons between the processes

4.3 WEKA – Text preprocessing tool:

Weka is a famous machine learning software written in java and developed at the University of Waikato. WEKA provides a large collection of machine learning algorithms for data preprocessing, categorization and classification.

Weka is an open source toolkit that allows users to extend it by adding new machine learning techniques. WEKA by itself does not support any Arabic techniques for text stemming; Saad and Ashour [58] implemented and integrated Khoja and Light 10 stemmers into weka. The researcher also integrated the proposed stemmer so Arabic users can use it and compare it to Khoja and Light 10 if needed. In the result section we will use the new Arabic IR tool for comparisons. Figure 4.20 shows the new algorithm added into WEKA.

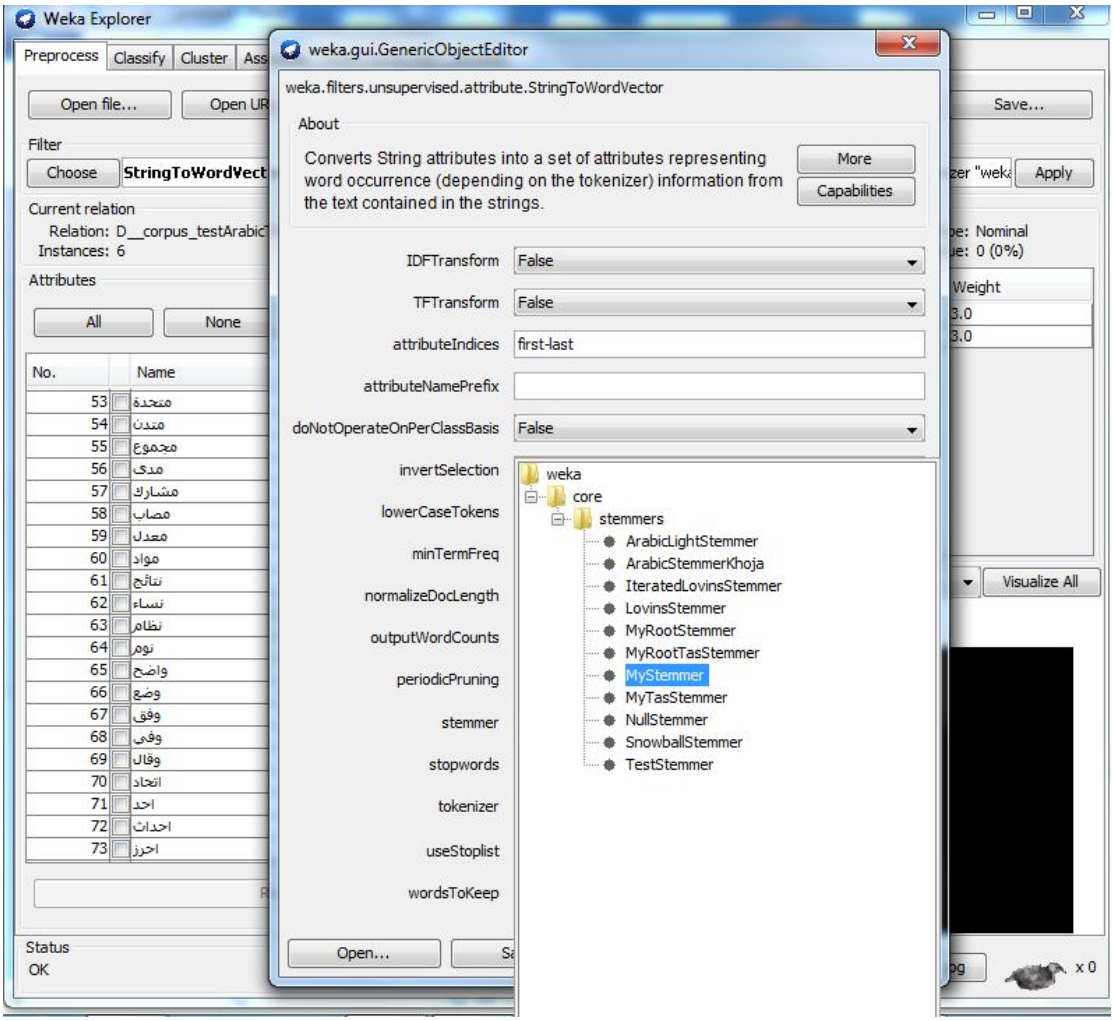


Figure 4.20: Weka Arabic Stemmers including the proposed root and light stemmers

Chapter 5. Experimental Results

The chapter will introduce the new proposed stemmer and shows how the stemmer has solved the problem of irregular words, broken plural patterns and blind removal of affixes, also the normalization and tokenization techniques will be introduced.

The proposed Arabic IR tool will be used to compare between the three stemming algorithms; Khoja, Light 10 and the proposed one. The comparison will be according to number of attributes, stemming time, and the effect of using the stemmer over text classification using naïve base multinomial and K-NN techniques. The effect of the stemmer on text classification can be measured by specifying the accuracy, precision, recall and time required for building models. WEKA will be used to compare the effect of using one of the three stemmers over text classification using support vectors machine classifier as it is not implemented in the proposed tool yet.

All results were generated using OSAC, CNN and BBC corpora as Arabic data set on 64-bit machine with 6GB RAM and core i5 processor. The new Arabic tool was developed using java programming language with JDK 1.6 and the open source external package "fchart" to draw the charts of results. We also used WEKA machine learning and "Apache ant" package for compiling WEKA after integrating the new stemmer within it.

5.1 Datasets specifications

This section describes and identifies the specifications of datasets used in all experiments over all algorithms. Datasets used are: CNN, BBC and the open source Arabic corpus OSAC collected by Saaed and Ashour [58] (available free for Arabic IR researchers).

5.1.1 CNN Corpus

The CNN corpus is selected from cnnarabic.com and includes 5070 text documents distributed over six classes (Business, Entertainments, Middle East News, Science & Technology, Sports and World News). The corpus contains 2241348 words. Table 5.1 describes the distribution of text documents over the six classes.

Table 5.1: Distribution of text documents over the six classes of CNN Corpus

#	Category	Number of documents
1	Business	836
2	Entertainments	474
3	Middle East News	1462
4	Science & Technology	526
5	Sports	762
6	World News	1010
Total		5070

5.1.2 BBC Corpus

The BBC corpus is selected from bbcarabic.com and includes 4763 text documents distributed over seven classes (Middle East News, World News, Business & Economy, Sports, International Press, Science & Technology, and Art & Culture). The corpus contains 1860786 words. Table 5.2 describes the distribution of text documents over the seven classes.

Table 5.2: Distribution of text documents over the seven classes of BBC Corpus

#	Category	Number of documents
1	Middle East News	2356
2	World News	1489
3	Business & Economy	296
4	Sports	219
5	International Press	49
6	Science & Technology	232
7	Art & Culture	122
Total		4763

5.1.3 Open Source Arabic Corpus - OSAC

OSAC corpus is collected from different web sites and includes 22429 text documents including CNN and BBC documents, each text document belongs to one of the ten classes (Economics, History, Education & Family, Religious and Fatwas, Sports, Health, Astronomy, Low, Stories, Cooking Recipes). The corpus contains about

18000000 words. Table 5.3 describes the distribution of text documents over the ten classes.

Table 5.3: Distribution of text documents over the ten classes of OSAC Corpus

#	Category	Number of documents
1	Economics	3102
2	History	3233
3	Education & Family	3608
4	Religious and Fatwas	3171
5	Sports	2419
6	Health	2296
7	Astronomy	557
8	Low	944
9	Stories	726
10	Cooking Recipes	2373
Total		22429

5.2 Tokenization and Normalization Effects

Normalization is essential for any kind of frequency-based analysis, so words such as "ألوان", "الـــوان", "ألــوان", "ألوان!" and "الوان" are not considered unique words. When dealing with human-generated text and typos, differences in presentation are bound to occur. Table 5.4 shows the effect of applying normalization on random sample selected from CNN corpus that includes punctuations, diacritics and other special characters.

Table 5.4: Effect of tokenization and normalization

Tokens	Proposed Tokenizer & Normalization	Light 10	Khoja
"امارات .امارات :امارات امارات، امارات"	امارات	"امارات .امارات :امارات امارات، امارات"	"امارات .امارات :امارات امارات، امارات"
(اللاعبين)	اللاعبين	(اللاعبين)	(اللاعبين)

Tokens	Proposed Tokenizer & Normalization	Light 10	Khoja
"متطرف متطرف)	متطرف	"متطرف متطرف)	"متطرف متطرف)
"000" "000"، "14 "26 "300"، ": "Goups"	NULL	"000" "000"، "14 "26 "300"، ": "Goups"	"000" "000"، "14 "26 "300"، ": "Goups"
/اغسطس اغسطس/آب .اغسطس/آب اغسطس/آب،	اغسطس اب	/اغسطس اغسطس/اب .اغسطس/اب اغسطس/اب،	/اغسطس اغسطس/آب .اغسطس/آب اغسطس/آب،

As illustrated in Table 5.4, the new algorithm has successfully removed all unnecessary characters that affect the IR process. The "\n" is used as a tokenizer with Light 10 and Khoja, while keeping the original normalization techniques for these algorithms which was described before in chapter 2. For our work, the regular expression tokenizer "\r\n\t.,;:\"()?!" has been used with the normalization process described in chapter 3.

Punctuations and special characters have major effects on IR, especially text classifications techniques. Figure 5.1 shows the effect of using normalization technique and regular expression as tokenizer in reducing the number of features.

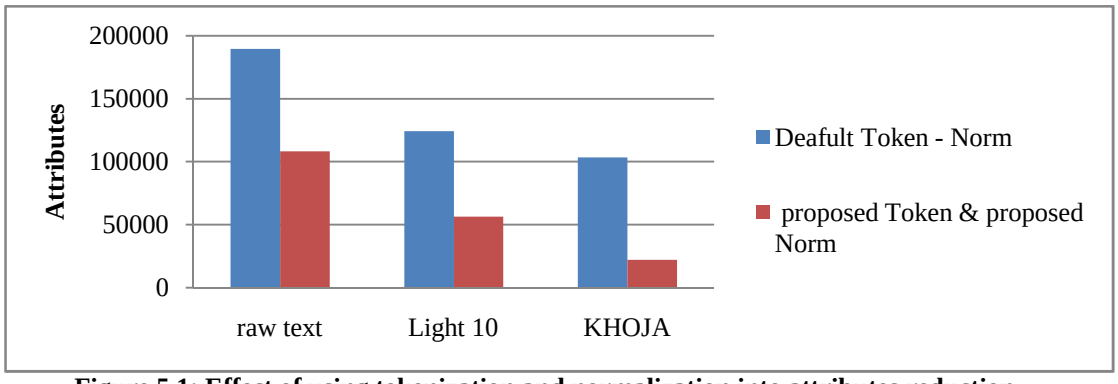


Figure 5.1: Effect of using tokenization and normalization into attributes reduction

The figure also shows the impact of using regular expression within predefined algorithms Khoja and Light 10.

Figure 5.1 introduces the effect of using tokenization and normalization in CNN corpus, the figure shows that using the new process will reduce the number of attributes of raw text from 189706 to 108208 words, which is about 43% reduction. Also, the new process has a great effect when using with Khoja and Light 10 as it reduces the number of attributes by 79% and 54% respectively. The huge reduction of attributes for Khoja depends on the steps of the algorithm itself for data after preprocessing, and because Khoja will return the root of the words so that many forms of the word will be returned as only one.

The impact of tokenization and normalization can be also recognized from the accuracy of text classification process, which is affected by the number of attributes that also affects the time for building models. Figure 5.2 shows the comparison between text classification accuracy with and without using new tokenization and normalization.

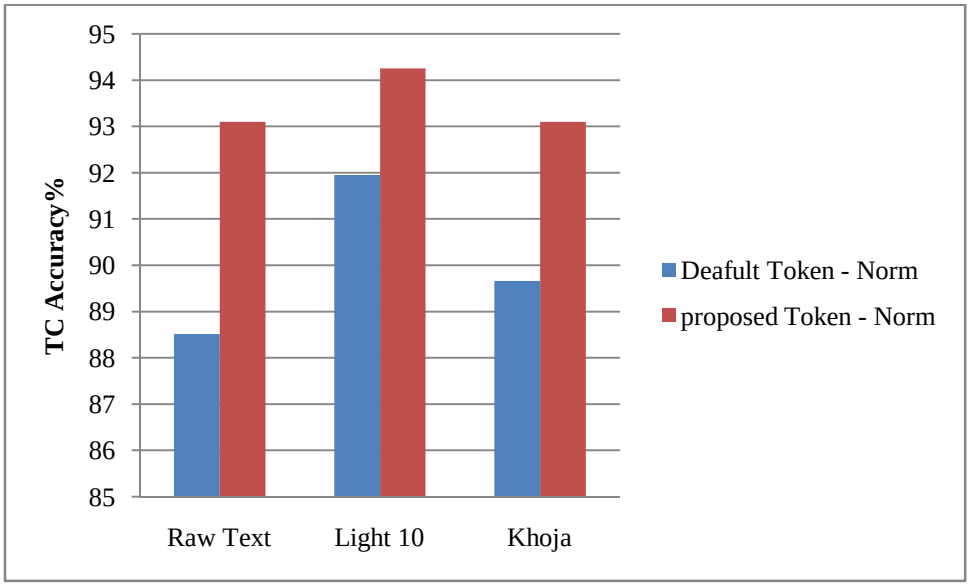


Figure 5.2: Text Classification accuracy using Naive Base Multinomial Classifier with normalization and without

The above figure shows the comparison between the text classification accuracy with and without normalizing data. The comparison is done on a sample of CNN corpus using Naïve Bayes Multinomial as text classifier and shows that using text preprocessing affects the classification process. So the researcher recommends using

text preprocessing before stemming, using regular expression in tokenization process and using these techniques in pre building stemming techniques like Khoja and Light 10.

5.3 Broken plurals and rule based effect

In Table 5.5 different words are presented, so it is noticeable how every algorithm we use deals with them to extract the stem word. In the proposed algorithm, we keep words that match patterns without stemming if that pattern is singular, but if the pattern is plural then the word is converted to the singular form.

Table 5.5: Word stemming comparison between the three algorithms

Word	Proposed stemmer	Light 10	Khoja
الوان	لون	وان	لون
مشاهير	مشهور	مشاهير	شهر
باستثناء	استثناء	باستثناء	ثني
اخبار	خبر	اخبار	خبر
اصوات	صوت	اصو	صوت
بسطاء	بسيط	بسطاء	بسط
اقوياء	قوي	اقوياء	قوا
ضحايا	ضحيه	ضحايا	ضحي
قلائل	قليل	قلائل	قلل
طوابير	طابور	طوابير	طوابير
وليفارق	يفارق	ليفارق	فرق
منظمات	منظم	منظم	ظمأ

Light 10 algorithm as shown from the result of table 5.6 was the most affected by broken plurals words, and as we can see from table 5.5 that words that match patterns like "مفاعيل", "فعلاء", "فعايا" will be returned as it is without conversion to singular form. Also problems may arise in some words, since original letters might be removed; this can be shown in the word "الوان"and "اصوات"as the letters "ال" and "ت" were removed respectively, these problems can be noticed when applying Khoja stemmer.

Table 5.6: purity result for three stemmers

		Euclidian	Cosine
Khoja	Purity	0.6	0.62
Light 10	Purity	0.52	0.54
Proposed Stemmer	Purity	0.7	0.72

Table 5.6 shows the result of using purity measurement, which are widely used to evaluate the performance of the classification process. The results shows that Light 10 affects negatively the clustering due to the ambiguity created when we applied the stemming.

So in order to increase the efficiency of the stemmer, the researcher recommends defining rules for removing affixes and using pattern matching process to match many forms of the word to the same stem.

5.4 Effects of Stemming in attribute reduction

Stemming is the process of redirecting the derived words to single form "stem". This process leads to gathering different forms of word into one form that leads to attribute reduction. Attribute reduction is important for information retrieval as it fasten the process of data retrieval and increases accuracy.

Figure 5.3 shows attribute reduction when using the three different types of stemmers the proposed stemmer, Light 10 and Khoja for the corpora CNN, BBC and OSAC. The proposed stemming reduction by 78-85 % with average of 81%, for Light 10 the reduction by 38-56% with average of 46%, while Khoja reduction by 47-70% of the raw text with average of 56%.

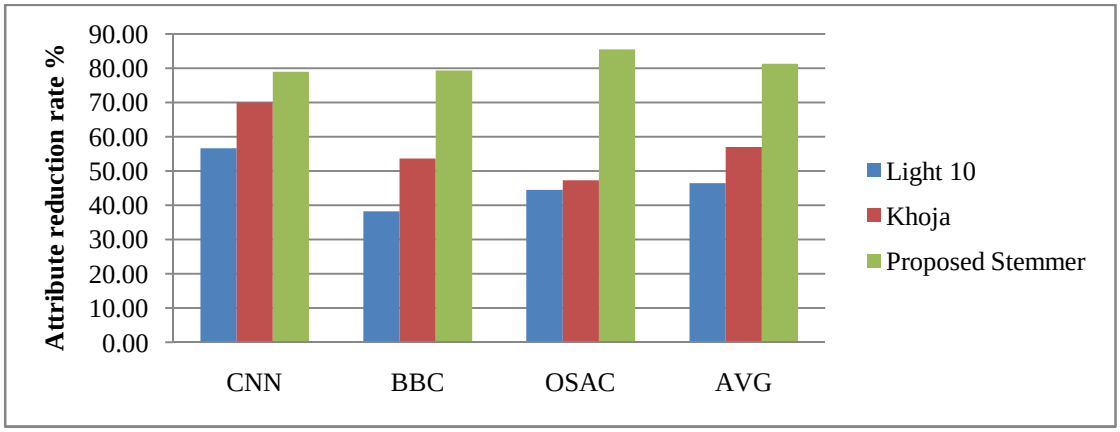


Figure 5.3: Attribute reduction rate using Light10, Khoja and Proposed stemmer

The new Arabic IR tool allows user to choose from different types of attribute reduction and merging techniques with new normalization/tokenization. Figure 5.4 shows the results of several integrations done using the tool. The results show that using Khoja with our new normalization techniques, regular expression as tokenizer and minimum time frequency equals to five has the highest attribute reduction, while using Khoja or Light 10 without the new normalization/tokenization techniques will always produce reduction average less than the proposed technique.

From Figure 5.4 we notice that applying normalization and stemming techniques with other feature selection techniques like minimum frequency time reduced the number of attributes and returned different forms of word to the same single form. This reduction is necessary to save memory, storage and time when applying information retrieval processes.

We can also notice that Khoja + minimum frequency of 5 words has the highest reduction rate. The order of reduction techniques from the highest to lowest reduction rate as shown in Figure 5.4 is: Khoja + min 5, Khoja + norm + min3, Khoja + norm + min5, proposed stemmer + norm + min5, Light10 + min5, proposed stemmer + norm + min3, Khoja + min 3, Light10 + norm + min5, Light10 + min3, Light10 + norm + min3, proposed stemmer + min3, proposed stemmer, Khoja, Light10, raw text.

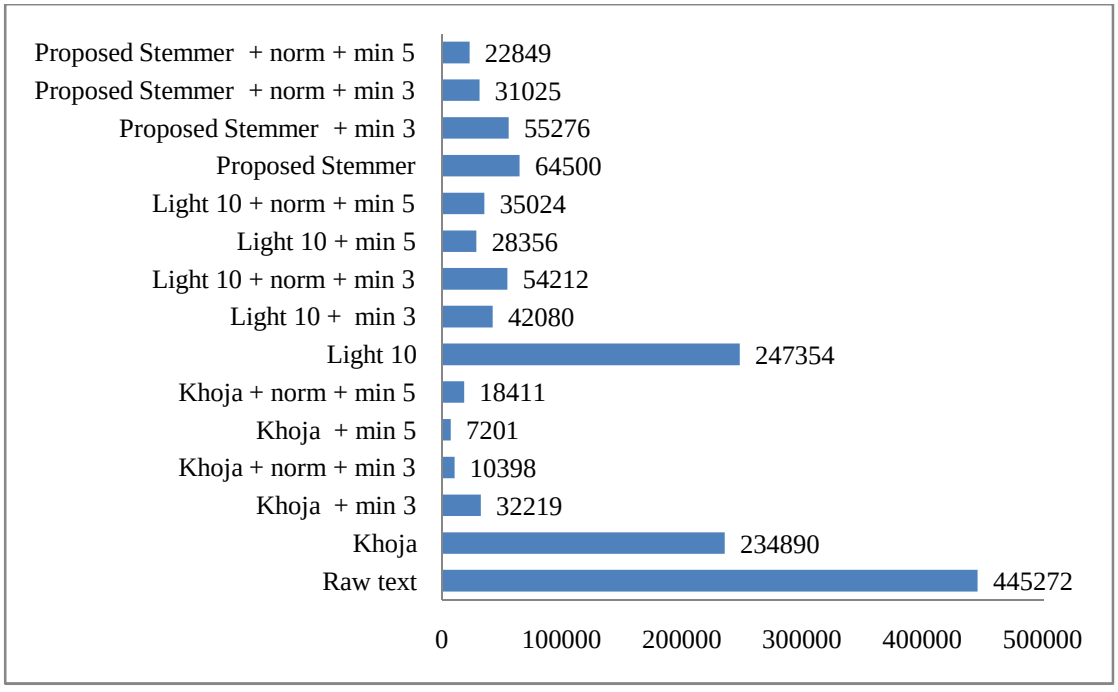


Figure 5.4: Attribute reduction using several techniques over OSAC Corpus

Despite the huge reduction of Khoja stemmer when used with normalization, the researcher recommends using the proposed stemmer as it is more proper than stemming from linguistics and semantic point of view, and faster than Khoja as it will be described in the next section.

5.5 Stemming Time

Figure 5.5 shows the time taken for each stemming approach. The time includes tokenization, normalization and stemming. Light 10 requires the least time as it just removes the affixes without checking the remaining word if it has meaning or not, while the process of comparing word according to rules will increase the time of the proposed stemmer. Khoja requires more time as it checks root against predefined root dictionary.

From Figure 5.5, the discussion in section 5.3 and despite the increase in stemming time for the proposed stemmer than Light 10, the researcher recommends using the proposed stemmer technique as it solves the problem of blind removal of affixes which produces words without meaning and it also solves the problem of broken plural.

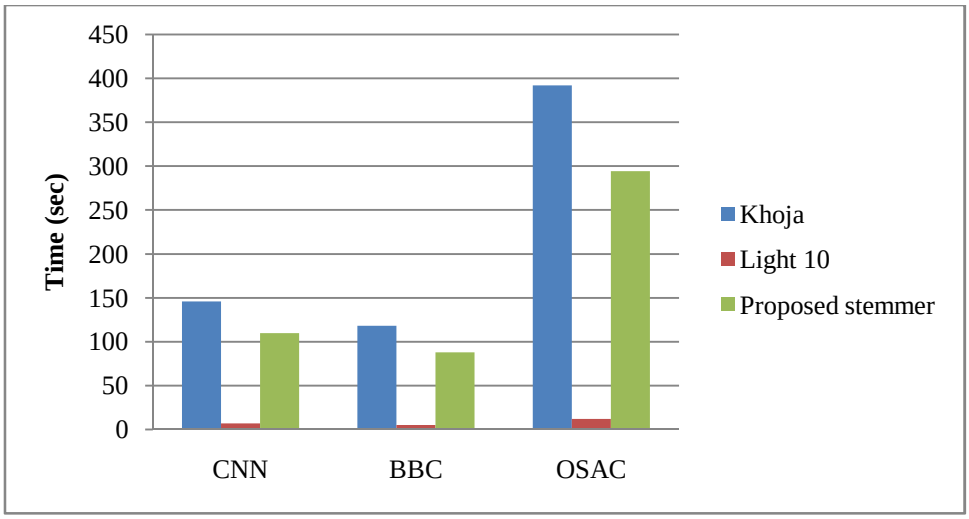


Figure 5.5: Khoja, Light 10 and Proposed stemming time for CNN, BBC and OSAC corpus

5.6 Effect of stemmers on classification accuracy

The impact of using the proposed stemmer vs. using Khoja or Light 10 on different corpora is depicted in Figure 5.6. The figure shows the classification performance when using each stemmer in CNN, BBC and OSAC corpus.

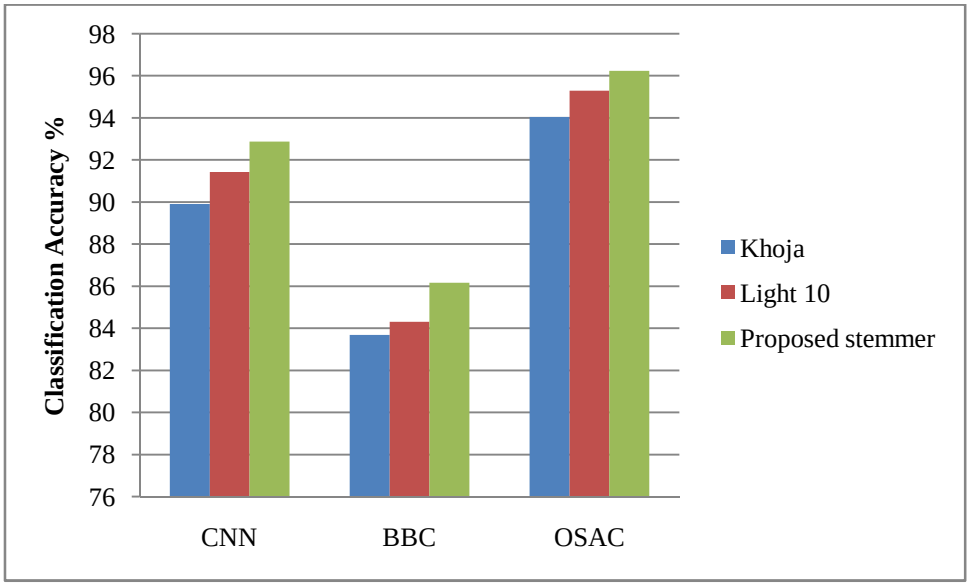


Figure 5.6: The accuracy of proposed stemmer vs. Khoja vs. Light 10 for different corpus

The classification was done using Naïve base classifier with word count and Min-F 5 as feature reduction technique. From the figure we can notice that the proposed stemmer leads to superior performance when used in all tested corpora.

Using the proposed stemmer will take more time in building model for classification than Khoja stemmer due to the high dimensionality. Figure 5.7 shows the time taken to build Naïve Bayes classifier model when using the proposed stemmer, Khoja and Light 10 in different corpora.

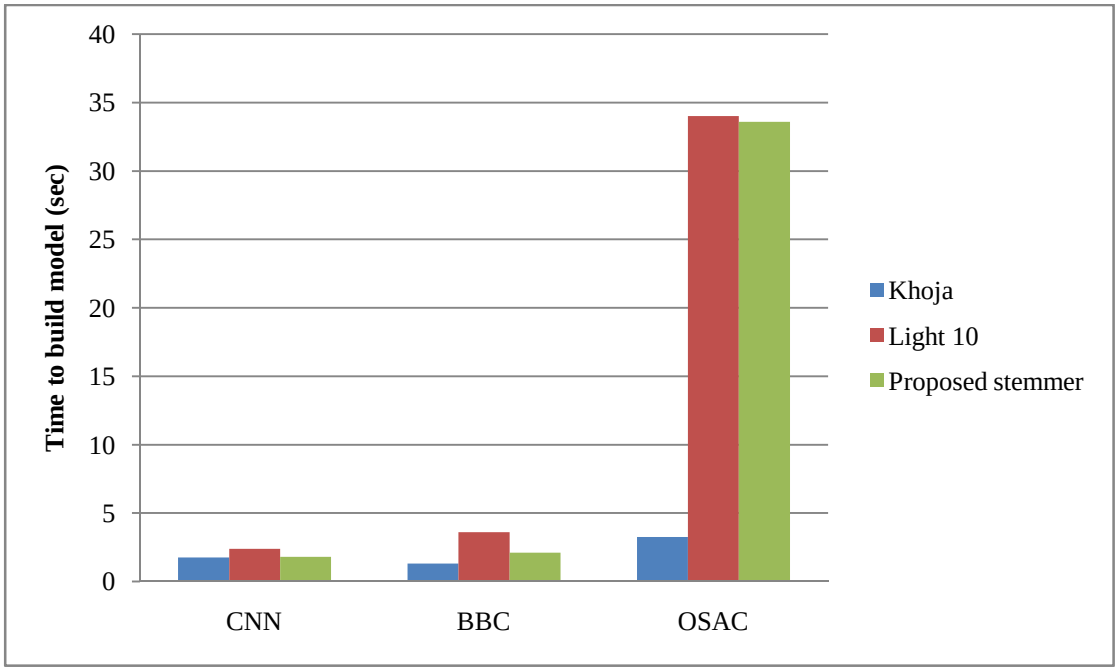


Figure 5.7: time taken to build model using proposed stemmer vs. Khoja vs. Light 10 for different corpus

The reason that Khoja has less time to build model than the proposed stemmer is that the number of attributes returned when using Khoja will be less than the ones returned by the proposed stemmer, as Khoja returned all words to the root form which allows more forms of the pattern to return to only one form. The researcher still recommends using the proposed stemmer over using Khoja as it is more accurate and leads to superior performance as shown in Figure 5.6.

Figure 5.8 shows the effect of using the proposed stemmer on recall and precision values. The proposed stemmer will increase the value of recall and precision to 87% and 92% respectively which is better than using Khoja or Light 10.

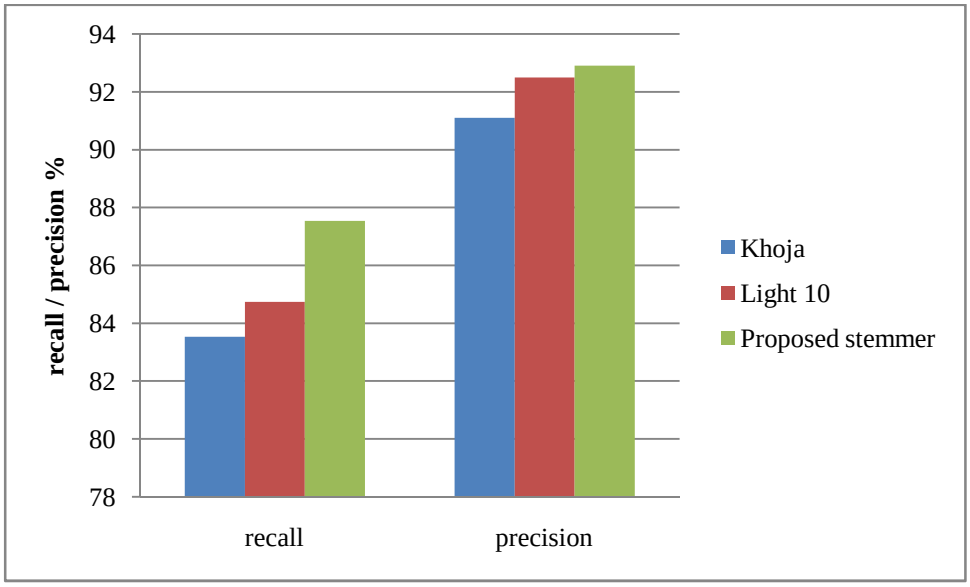


Figure 5.8: Average recall and precision using Khoja, Light 10 and the proposed stemmer

The reason that the precision rate is greater than recall is that the data is not distributed equally to the classes, and that the selection of training data depends on the number of files for each class.

The effect of selecting training data randomly is shown in Figure 5.9. The figure shows the accuracy of classifying data using nine random selections of 66% from data to be training data, while keeping the rest for testing. The result – 5.9 - shows that the accuracy will range between 91.3% and 92.9 % with average 91.85%.

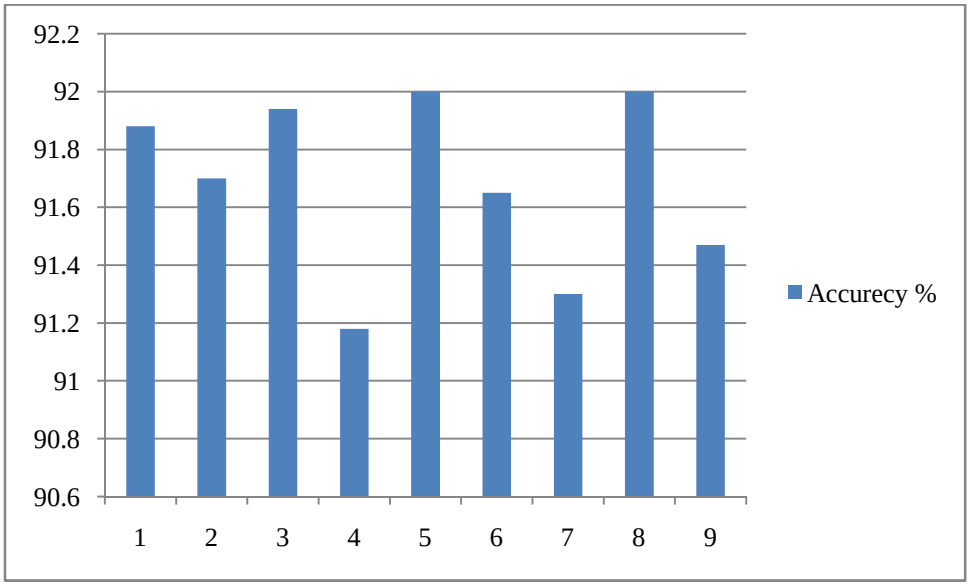


Figure 5.9: Accuracy using random selection of training data with the proposed stemmer

As the K-NN classifier depends on measuring distances between objects to classify it's, so K-NN is sensitive for weighting techniques and will be used to show the effect of term weighting schemes; the result is shown in Figure 5.10.

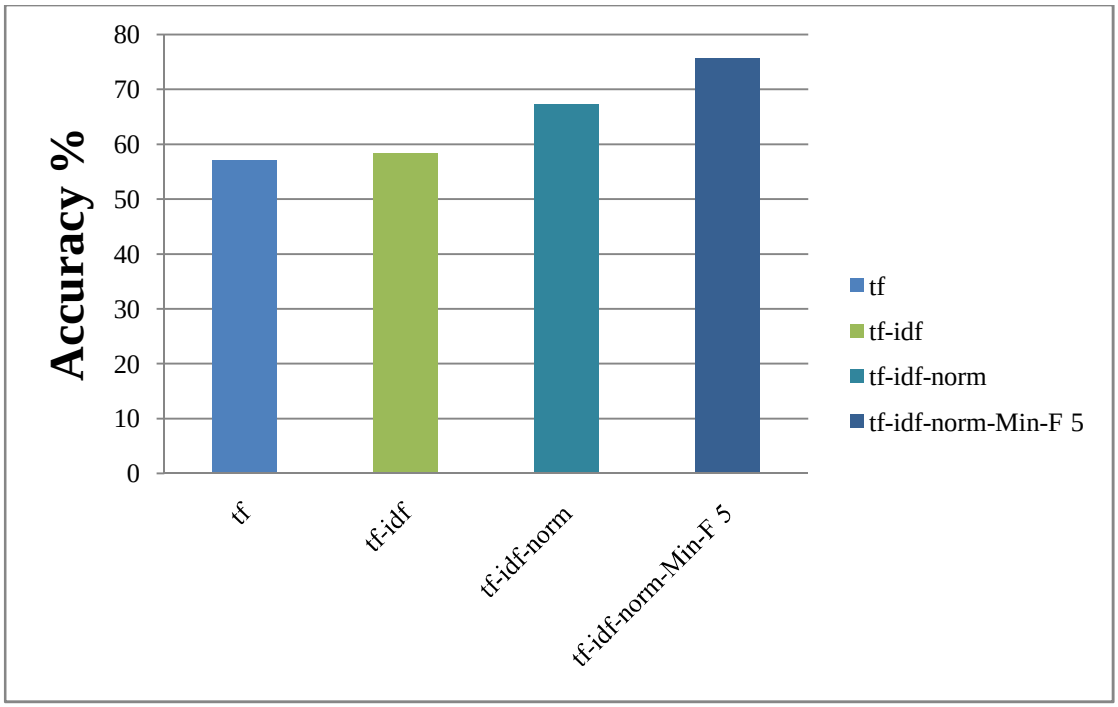


Figure 5.10: The effect of using different term weighting frequency with K-NN on accuracy

Figure 5.10 shows that using tf-idf with normalization and minimum frequency 5 leads to the highest accuracy rate as the normalization process aims to reduce the distances between objects. The test was done using k = 11 and cosine similarity as distance measurement.

Chapter 6. Conclusion

6.1 Conclusion

This Thesis presents a proposed Arabic stemmer and a new Arabic IR tool to test the stemmer and compares the result with other previous stemmers. The proposed stemmer is added to one of the most famous IR platform WEKA to help researchers focus on using the features of this platform for researching and improving Arabic IR filed.

The research in this thesis focused on the different phases of stemming techniques, discussed the strengths and weaknesses of the existing approaches and introduced new method to improve performance.

The researcher improved the tokenization process by using regular expression as it has good impact on removing special characters and punctuations. The researcher also mentioned that there is no standard used in the normalization process, so the researcher collected and analyzed the normalization techniques used in different previous stemmers and gathered them into eight steps; the user then can select from these steps from the GUI of new Arabic IR tool.

A new hybrid stemmer that depends on using pattern matching and affixes removal was proposed. The new stemmer solves the problems of broken plurals, blind affixes removal and irregular words that previous algorithms suffered from. By using the proposed stemmer there is no need for predefined lists of irregular words as it is detected automatically when compared with rules.

Finally, a new Arabic IR tool has been developed which has many options, it allows user to load any data set, choose one of three included stemmers, choose from the eight normalization steps, define the set of constants like “prefixes, suffixes, stopwords”, text classification, make comparisons between stemmers and extract charts that show these comparisons.

The experimental results show that the proposed stemmer has great effect in text classification fields and gives better performance depending on many factors, such as preprocessing, feature selection method and classification method.

Using normalization is an important step before stemming, as it reduces the number of attributes by removing unnecessary characters and words, and increases efficiency of the stemmer. Normalization reduces the number of attributes to about 50% and increases text classification accuracy by about 6%.

The problem of broken plurals affects the stemming process, and must be solved. Stemming irregular words will lead to words without meaning, and will change the structure of the word, so it must be returned without stemming.

The researcher recommends using light stemmer, as it is more proper than root stemmer from linguistics and semantic point of view, and grants more accurate results when used in IR applications. Using the proposed stemmer increases accuracy of text classification to an average of 91.7% which is better than using Light 10 or Khoja which achieve an average accuracy of 90.2 % and 89.17% respectively.

6.2 Future Work

In the future work, we shall work on extending the new IR tool to include more stemmers, weighting techniques and classification techniques that allow researchers to make more accurate decisions when analyzing and comparing techniques.

We shall define an Arabic dataset that can be used for testing any stemmer by defining queries, so the stemmers can be compared together according to the results of those queries.

We also need to define a set of rules that allow the broken plural pattern to be converted only to the true singular form; this may be achieved by measuring the similarity between the word and each singular form then selecting the most similar one.

References

[1] Prabhakar Raghavan and Hinrich Schütze Christopher D. Manning, *Introduction to Information Retrieval*, 2nd ed. Cambridge: Cambridge University Press, 2009.

[2] V. HARSHAVARDHAN and J. KUMAR K JAWAHAR BABU, "The Role Of Information Retrieval In Knowledge," *International Journal of Social Science & Interdisciplinary Research*, vol. 1, no. 10, 2012.

[3] G. Salton, *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., 1989.

[4] Ballesteros and Connell Larkey, "Improving stemming for Arabicinformation retrieval: Light stemming and co-occurrence analysis," in *ACM 2002 Article. Bibliometrics Data Bibliometrics*, Tampere, Finland, 2002.

[5] Perce, Tarry and Willett Lennon, "An evaluation of some conflation algorithms for information retrieval," *Taylor Graham Publishing London*, vol. 3, no. 177-183, pp. 230-236, 1988.

[6] S. and Garside Khoja, "R. Stemming Arabic Text," *Computing Department, Lancaster University, Lanca*, pp. 1-7, 1999, [Available at: http://www.comp.lancs. ac.uk/computing/users/khoja/stemer.ps].

[7] Croft and C. Van Rijsbergen, "An evaluation method for stemming algorithms," in *SIGIR '94 Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, London, 1994, pp. 42-50.

[8] D. Hull, "Stemming algorithm – a case study for detailed evaluation," *Journal of the American Society for Information Science*, vol. 47, no. 1, pp. 70-84, 1996.

[9] W. and Pohlmann, R Kraaij, "Viewing stemming as recall enhancement," in *SIGIR '96 Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, 1996, pp. 40-48.

[10] Wikipedia. (2014) Wikipedia. [Online]. http://en.wikibooks.org/wiki/Arabic

[11] Nizar Habash, Owen Rambow and George Kiraz, "Morphological Analysis and Generation for Arabic Dialects," in *Semitic '05 Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages* , Columbia, 2005, pp. 17-24.

[12] Marwan Bawab, "Arabic language processing in information systems," , 2004.

[13] Mahmoud Gaatar Jane Wightwick, *Arabic verbs and essentials of grammars*, 2nd ed.: McGraw-Hill, 2007.

[14] S., Nagi, M., Adly N. Al-Ansary, *Towards analyzing the International Corpus of Arabic:*

Progress of Morphological Stage, 3rd ed.: Bibliotheca Alexandrina, 2008.

[15] M, Atwell, E. Sawalha, "Comparative evaluation of Arabic language morphological analyzers and stemmers," *International Journal of Managing Information Techn*, vol. 5, no. 2, May 2013.

[16] A. Naamany and A. Z. A. Bakar M. R. Al-Maimani, "Arabic information retrieval: techniques, tools and challenges," , Dubai , 2011, pp. 541 - 544.

[17] Robert Krovetz, "Viewing morphology as an inference process," 1993.

[18] R. Al-Shalabi and Evens, "A computational morphology system for Arabic," *Association for Computational Linguistics*, pp. 66-72, 1998.

[19] G. Kanaan and H. Al-Serhan R. Al-Shalabi, "New approach for extracting Arabic roots," *International Arab Conference on Information Technology* , pp. 42-59, 2003.

[20] Rania Elkhoury and Jeffrey Coombs Kazem Taghva, "Arabic Stemming Without A Root Dictionary," in *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference*, vol. 1, Las Vegas, 2010, pp. 152-157.

[21] Leena Lulu, Belkhouche and Saad Harous Anas Boubas, "A Novel Approach for an Arabic Stemmer Using Genetic Algorithms," , Dubai, 2011, pp. 77-82.

[22] Dr. Raed Kanaan and Dr. Ghassan Kanaan, "AN IMPROVED ALGORITHM FOR THE EXTRACTION OF TRILITERAL ARABIC ROOTS," *European Scientific Journal*, vol. 10, no. 3, January 2014.

[23] Qussai Yaseen andI smail Hmeidi, "Extracting the roots of Arabic Words without removing affixes," *Information Science & Library Science*, 2014.

[24] Aljlay and Frieder, "Improving the retrieval effectiveness via light stemming approach," *Information and knowledge management*, pp. 340-347, 2002.

[25] Al-Ketbi O. Shaikha, Al-Kaabi A. Amna and Al-Shebli Al-Ameed k. Hayder, "Arabic Light Stemmer: A new Enhanced Approach," *The second international conference on innovations technology*, 2005.

[26] Y. Alhanini and M. J. A. Aziz, "The Enhancement of Arabic Stemming by Using Light Stemming and Dictionary-Based Stemming," *Computer Science & Communications* , vol. 4, no. 9, Septemper 2011.

[27] S.M.M Tahaghoghi, Falk Scholer Nwesri A., "Stemming Arabic Conjunctions and Prepositions," *String Processing and Information Retrieval, Lecture Notes in Computer Science*, vol. 3772, pp. 206-2017, 2005.

[28] Nwesri A., S.M.M. Tahaghoghi and Falk Scholer, "Arabic Text Processing for Indexing and Retrieval," *String Processing and Information Retrieval*, 2007.

[29] Behloul A, Dilekh T., "Implementation of a New Hybrid Method for Stemming of Arabic Text," vol. 46, no. 8, 2012.

[30] Tashaphyne. (2010) Arabic light stemme. [Online]. http://tashaphyne.sourceforge.net

[31] Jian-Yun Nie Youssef Kadri, "Effective Stemming for Arabic Information Retrieval," *THE CHALLENGE OF ARABIC FOR NLP/MT*, 2006.

[32] Chen A. and Gey F., "Building an Arabic stemmer for information retrieval," *School of Information Management and Systems. University of California at Berkeley*, 2011.

[33] Suleiman H. Mustafa, "Word Stemming for Arabic Information Retrieval: The Case for Simple Light Stemming," *ABHATH AL-YARMOUK: "Basic Sci. & Eng."* , vol. 21, no. 1, pp. 123-144, 2012.

[34] Djelloul Ziadi, Hadda Cherroun and Younes Guellouma Attia Nehar, "An efficient stemming for Arabic Text Classification," in *Innovations in Information Technology (IIT)*, Abu Dhabi, 2012, pp. 328-332.

[35] Boreham and G. Adamson, "The Use of an Association Measure Based on Character Structure to Identify Semantically Related Pairs of Words and Document Titles," *Information Storage and Retrieval*, vol. 10, no. 7, Augest 1974.

[36] EW De Luca and A Nürnberger F Ahmed, "Revised n-gram based automatic spelling correction tool to improve retrieval effectiveness," in *Information Retrieval and Natural Language Processing*.: Computer Science and Computer Engineering with Applications, 2009, vol. 40, pp. 39-48.

[37] Lachkar and Ouatik Froud, "Stemming Versus Light Stemming for Measuring the Simitilarity between Arabic Words with Latent Semantic Analysis Model," 2012.

[38] M. K. Saad and W. Ashour, "Arabic Morphological Tools for Text Mining," *6th International Conference on Electrical and Computer Systems*, 2010.

[39] K. Darwish, "Probabilistic methods for searching OCR-degraded Arabic text," *Doctoral Dissertation- Unpublished Ph.D. Thesis*, 2003.

[40] D., Wanas, N.M., Darwish, N.M., and Hegazy, N. Said, "A Study of Text Processing Tools for Arabic Text Categorization," *Electronics Research Institute, Cairo, Egypt* , 2009.

[41] Masnizah Mohd Qusay Walid Bsoul, "Effect of ISRI Stemming on Similarity Measure for Arabic Document Clustering," *7th Asia conference on Information Retrieval Technology* , pp. 584-593, 2011.

[42] B. Al-Salemi and M. J. Ab Aziz, "Statistical Bayesian Learning for Automatic Arabic Text Categorization," *Journal of Computer Science*, vol. 7, no. 1, 2011.

[43] A. N. & Al-Fares, W. De Roeck, "A morphologically sensitive clustering algorithm for identifying Arabic roots," *Proceedings*, 2000.

[44] Arafa and Darwish El-Disooqi, "Stemming techniques of Arabic Language: Comparative Study from," *ISSR Cairo University*, 2009.

[45] Dunham M., *Data mining: Introductory and advanced topics*, 1st ed.: Pearson Education, 2003.

[46] Duwairi R., "Machine Learning for Arabic text Categorization," *Journal of the American Society for Information Science and Technology*, pp. 1005-1010, 2006.

[47] Sanger J. Feldman R., "The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data," *Cambridge University Press*, 2007.

[48] Bensaid A., Rachidi T. El-Kourdi M., "Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm," *Computational Linguistics*, 2004.

[49] Al-Shalabi R., Ghwanmeh S. Kanaan G., "A comparison of text-classification techniques applied to Arabic text," *Journal of the American Society for Information Science and Technology*, 2009.

[50] Kannan G.and Gharaibeh H. Al-Shalabi R., "Arabic text categorization using KNN algorithm," *computer science and information technology CSIT06*, 2006.

[51] El-Halees A., "A Comparative Study on Arabic Text Classification," *Egyptian Computer Science Journal*, 2008.

[52] Hadi W. Musa, Al-shammare G. Thabtah F., "VSMs with K-Nearest Neighbour to Categorize Arabic Text Data," 2008.

[53] Changsheng Yang, and Benjamin Wong. Gerard Salton, "A vector-space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 1, 1975.

[54] I. A. El-Khair, "Effects of stop words elimination for Arabic information retrieval: a comparative study," *International Journal of Computing & Information Sciences*, vol. 4, no. 3, pp. 119-133, 2006.

[55] and Line Eikvil Kjersti Aas, "Text Categorisation: A Survey," *Technical report, Norwegian Computing Center*, 1999.

[56] NIGAM and MCCALLUM, "Text Classification from Labeled and Unlabeled Document using EM," *Kluwer Academic Publishers, Boston. Manufactured in The Netherlands*, pp. 1-34.

[57] S. Kalyani and K.S. Swarup, "Particle swarm optimization based K-means clustering approach for security assessment in power systems," vol. 30, 2011.

[58] Motaz K. Saad. (2010) Open Source Arabic Language and Text Mining Tools. [Online]. http://sourceforge.net/projects/ar-text-mining