A Novel Rule-Based Root Extraction Algorithm for Arabic Language

Nisrean Thalji[1], Nik Adilah Hanin[2]
Department of Computer Engineering, School of Computer and Communication Engineering
University Malaysia, Perlis, Malaysia

Sohair Al-Hakeem[4]
Computer Science Department
Ajloun National Univesity
Ajloun, Jordan

Walid Bani Hani[3]
Department of Computer Science
Higher Colleges of Technology
Ras Al Khaimah, United Arab Emirates

Zyad Thalji[5]
Department of Management Information System
Imam Abdulrahman Bin Faisal University
Kingdom of Saudi Arabia

***Abstract*—Non-vocalized Arabic words are ambiguous words, because non-vocalized words may have different meanings. Therefore, these words may have more than one root. Many Arabic root extraction algorithms have been conducted to extract the roots of non-vocalized Arabic words. However, most of them return only one root and produce lower accuracy than reported when they are tested on different datasets. Arabic root extraction algorithm is an urgent need for applications like information retrieval systems, indexing, text mining, text classification, data compression, spell checking, text summarization, question answering systems and machine translation. In this work, a new rule-based Arabic root extraction algorithm is developed and focuses to overcome the limitation of previous works. The proposed algorithm is compared to the algorithm of Khoja, which is a well-known Arabic root extraction algorithm that produces high accuracy. The testing process was conducted on the corpus of Thalji, which is mainly built to test and compare Arabic roots extraction algorithms. It contains 720,000 word-root pairs from 12000 roots, 430 prefixes, 320 suffixes, and 4320 patterns. The experimental result shows that the algorithm of Khoja achieved 63%, meanwhile the proposed algorithm achieved 94% of accuracy.**

***Keywords*—Root; stem; rules; affix; pattern; corpus**

I. Introduction

Arabic texts are mainly categorized into two types. The first type is known as Classical Arabic e.g. the Qur'an text. The second type is called Modern Standard Arabic (MSA), which is the form that is used in all Arabic-speaking countries in publications, media and academic institutions [1]. The Modern Standard Arabic is then classified into three types, which are fully vocalized like elementary textbooks, partially vocalized like newspapers, and the non-vocalized text.

Vowels are used in Arabic to ensure the reading and the exact meaning of the words. If the word is non-vocalized, in many cases, it will represent an ambiguous word, and then we need to read the full sentence and sometimes the whole article or document to understand the exact meaning.

Root extraction is the process of extracting the root of the word. Root extraction correlates several terms into one common representation. Therefore, those words which are derived from the same root are grouped together. For example, root extraction algorithms reduce the word "fishing", "fished", and "fisher" to "fish". Root extraction is used in information retrieval systems, indexing, text mining, text classification, data compression, spell checking, text summarization, question answering systems and machine translation [2].

Arabic dialect contrasts from the Indo-European dialects morphologically, semantically, and grammatically. Building an Arabic root extraction is more complicated than building root extraction in any other European language such as English. English language root extraction is only concern with the removal of prefixes and suffixes [3].

Affixes in Arabic are prefixes, suffixes and infixes. Prefixes are attached at the beginning of the words, where suffixes are attached at the end, and infixes are found in the middle of the words [4]. For example, the word كبيوتكم which meaning is "like your houses" in English, ك is the prefix, which is a connected preposition, كم is the suffix, which is the subject here, and و is the infix. So, the root is بيت, where in English the preposition and subject are written separately. So, for the "houses" word no prefix, no infix, "s" is the suffix, and the root is "house".

In Arabic, words are made from roots and patterns. Patterns are non-consonant letters groupings which can be interceded on as templates [5]. Patterns can be added to the root of the word or can be found within the roots of the word following well-defined models [6]. Many words have the same pattern. The root of any words can be easily extracted if the word and the pattern are known. For example, if the words واستبدلته, واستجبرته, واستجهرته have the pattern of واستفعلته, therefore the roots will be بدل, جبر, جهر respectively.

As a result of a thorough investigation of existing algorithms, in this work, a new rule-based Arabic Root Extraction Algorithm (AREA) is proposed. Our algorithm is an extensive enhancement and improvement work which is done to overcome the limitations of the previous works that can be used in both IR and NLP applications in an effective way.

This paper is organized as follows. In Section II, the discussion regarding previous studies and their drawbacks is presented. Section III describes the proposed methodology, including details of each process. Section IV explains the experimental implementation of our algorithm and the evaluation process. Section V concludes the main points of the paper and gives some future directions.

II. Previous Studies

AREA can be categorized into a database search approach, statistical based approach, and a rule-based approach [7].

A. Database Approach

Database search approach is the simplest strategy; it simply looks for the root of the word in the lookup table. The database would also include a list of patterns that match different Arabic words and can be used to help identify different roots.

Most well-known works using this approach are Al-Fedaghi with Al-Anzi algorithm [8], and Al-Shalabi algorithm [9]. They proposed an algorithm to generate the root and pattern of a given Arabic word. The main problem of this type is when there is no pattern or root is matched from the database. The limitation of this method is the need to constantly update the database. Also, there is a possibility that the algorithm will detect more than one pattern for certain words.

B. A Weight-Based Approach

With this approach, the algorithm assigns different weights to letters in the word, and then, using mathematical calculations to find the root. Al-Serhan, Al Shalabi and Kannan algorithm [10] is an example of this approach. The main problem of this algorithm is it gives the same priority for the extra letters as the original letters. For example, it gives the same priority to (ف, ك, ب, ط, د) with (ص,ش , ز , ر, ذ, خ, ح,ج ,ث ض,), although these letters sometimes are not the original root letters. For example, if a word contains the letters (ف, ب) as a prefix or the letter (ك) as a suffix, the algorithm fails to identify the root. This happens when it gives the letters' root less priority than other letters in the word. For example, if the letters' roots in (أ, م, ن,ا) and the extra letters are in (ل,س,ه).

C. A Rule-Based Approach

Most of the AREA in the literature today are rule-based. In the rule-based approach algorithms, a set of rules are built to find the Arabic root from the original word. In most cases, this approach will also use a database of patterns and affixes as well. These algorithms affected by the way the rules are arranged as well as the number of rules. Such algorithms would also involve a pre-processing to find a possible root.

Khoja and Garside algorithm [11] is the most popular rule-based Arabic root extraction algorithm. Khoja and Garside algorithm reported 96% accuracy of their algorithm using newspaper text.

Al-Shalabi [12] presents Arabic root extraction algorithm, which is a rule-based algorithm that is used to extract trilateral roots of Arabic words. This algorithm has been tested on a corpus of 72 abstracts, 10582 words from the Saudi Arabian National Computer Conference and they achieved 92% of accuracy.

Another work, Al-Kabi and AL-Mustafa algorithm [13] is based on affix removal. They tested their algorithm on small data sets containing 1,827 words. The system unable to analyse 55 words, since their patterns are unknown. This failure mostly due to foreign (Arabized) words. The system is able to analyse the rest (1,772 words), but it was stated that the accuracy of extracting the right roots is 91%.

Sonbol, Ghneim and Desouki algorithm [14] is another rule-based root-extraction algorithm where the principal idea is based on the encoding of Arabic letters with a new code that preserves morphologically useful information and simplifies it's capturing toward retrieving the root. They conducted their experiments using two different corpuses. The first corpus consists of lists of word-root pairs (167162 pairs). The second corpus is a collection of 585 Arabic articles from different categories (policy, economy, culture, science and technology, and sport). This corpus consists of 377793 words. Overall, the algorithm yields about 96%-98% of accuracy.

Ghwanmeh, Al-Shalabi, Kanaan, Khanfar and Rabab'ah algorithm [15] proposed a rule-based algorithm to find trilateral Arabic roots. According to Ghwanmeh et al, their algorithm only unable to analyse words that are normally foreign, irregular, or do not have trilateral roots. A corpus of 242 abstracts from the Proceedings of Saudi Arabian National Computer conferences in machine-readable form is used in the testing procedure. The set of abstracts was chosen randomly from the corpus for analysis. The results obtained showed that the algorithm extracts the correct roots with an accuracy rate up to 95%.

Up until now, various rule-based algorithms have been proposed such as the Kchaou and Kanoun algorithm [16], El-Defrawy, El-Sonbaty, and Belal algorithm [17], and Ayedh and Guanzheng algorithm [18] and many more works[19] [20] [21] [22] [23].

III. Method

This section describes the methodology for the new Arabic root extraction algorithm. The presented algorithm will find all possible roots for each word. The root is the base form of the word that gives the main meaning of the word.

A. Normalization

Normalization is the process that leads to the removal of unwanted letters, punctuations, and non-letters. The normalization steps consist of the followings:

- Remove kasheeda symbol ("-").
- Remove punctuations.
- Remove diacritics.
- Remove non-letters.
- Replace Hamza's forms ء, ,آ ,ؤ ,إ ئ with أ .
- Duplicating any letter that has the Shaddah: "ّ" symbol.

B. Extracting the Constant Letters from the Word

The proposed algorithm finds all the possible roots of the word without removing prefixes and suffixes. It starts by extracting the constant letters in a word by applying the rules in the Table I. The starting process of the presented algorithm differs from most of the previous algorithms, because it does not start removing prefixes and suffixes from the words' derivations. Particularly, removing prefixes and suffixes from the words' derivations leads to omitting many letters from the root which leads also to wrong results. Most of the previous algorithms remove the prefixes and suffixes from the words' derivations which is depends on the expectation' processes. In other words, most of the previous algorithms do not sure exactly that prefixes and suffixes are affixes or not. For instance, consider the word " استماع". Most of the previous algorithms remove the prefix "است" from the word because they depend on the expectation' processes that the prefix "است" is founded in their prefix's lists. As a result, they remove it directly.

Next, we categorize the Arabic letters into groups as the work of Sonbol's Arabic root extraction algorithm. In Arabic, letters are categorized into two main groups; Constant and Nonconstant letters. Constant letters are: 'ث', 'ج', 'ح', 'خ', 'د', 'ذ', 'ر', 'ز', 'ش', 'ص', 'ض', 'ظ', 'ط', 'ع','ق', 'غ". If these letters appear in the derivation word, it also should appear in its root. For instance, the word "الجاحدون" has "جـ, حـ, د" constant letters. These constant letters must be part of the root. Therefore, constant letters are not being considered as affix letters.

The second Arabic letters' classification is the Non-constant letters which are divided into five categories; the prefix letters {ب, ف ،س ، ل}, the suffix letter {هـ}, the prefix-suffix letters {ك, م, ن}, the uncertain letters {ا, و, ي, ت} and an extra letter "ة". We face many urgent issues that need more understanding than constant letters' work because constant letters may appear in the derivation words, but not appear in their root.

TABLE I. RULES OF EXTRACTING THE CONSTANT LETTERS IN THE WORD

No	Rules	Example
1	Find out the constant letters in the word. If the number of constant letters is more than one letter, then they will be considered as one of the expected roots.	The input word "التقارير", the constant letters are {ق, ر, ر}, then {قرر} is one of the possible roots for the word "التقارير".
2	Check Ebdal rules to minimize the constant letters.	The input word "اصطحب", the constant letters are {ص, ط, ح, ب}, after applying Ebdal rules the constant letters become {ص, ح, ب}

C. Converting the Non-Constant Letters to the Constant Letters

The Non-constant letters in the derivation's word are the original root letters in some cases and considered as the additional letters to the root in other cases, depending on the position of the letters. In this section, a certain set of rules are applied to each letter in the non-constant letters' group in order to convert these letters into constant letters.

1) The prefix letters {ل, س, ف, ب}: The Prefix letters {ل, س, ف, ب} are one of the non-constants' letters. They are attached at beginning of the words. A certain set of rules has been implemented on each letter on the prefix letters' list to convert these letters from non-constant letters to a constant letter.

a) Prefix letter ل

Initially, the letter {ل} is a non-constant letter. It can be converted to a constant letter by applying the following rules:

Rule1: If the letter "ل " exists after the first constant letter, then the letter "ل" is treated as a constant letter. For example, with the word "اعتقل", the letters {ع, ق} have been identified constant letters. And the letter "ل" exists after the first constant letter. So, the letter "ل" is treated as a constant letter. Then the constants' letters list becomes {ل, ق, ع}.

Rule2: Check the position of the letter "ل" in the word. If the letter "ل" exists in the second half of the word, then it is treated as a constant letter. For example, consider the word "استلم". The letter "ل" is positioned in the second half of the word. Thus, in this case, it is considered a constant letter.

Rule3: If the letter "ل" is preceded by the letters "ال", it is treated as a constant letter. As it is in the word" الليل"

Rule4: The letter "ل" is treated as a constant letter if it has been preceded by one of these letters "ك, س, ي, م, ه, ت, ن". As it is in the following words" كلمس, يلمس, نلمس, تلمس, ملاك, هلاك, سلوك".

b) Prefix letter "س"

Initially, the letter "س" is a non-constant letter. It can be converted to a constant letter by applying the following rules:
Rule1: If the letter "س" exists after the first constant letter, the letter "س" is treated as a constant letter. For example, with the word "أجناس", the letter "ج" has been identified a constant letter. Letter "س" exists after the first constant letter. So, "س" is treated as a constant letter. The constants letters list becomes "ج, س".

Rule2: If the letter "س" is preceded by the letters "ال", it is treated as a constant letter. As it is in the word" السباع".

Rule3: The letter "س" is treated as a constant letter if it has been preceded by one of the letters "ل, ب, س, ك, هـ". As it is in the words" لسماع, بسماع,"

Rule4: The letter "س" is treated as a constant letter if it hasn't been followed by one of the letters "أ, ا, ن, ي, ت". As it is in the words "سلالم, سكان".

Rule5: Check the position of the letter "س" in the word. If the letter "س" exists in the second half of the word, it is treated as a constant letter. For example, with the word "ميؤوس", the letter "س" is positioned in the second half of the word. Thus, in this case, it is considered a constant letter.

Rule6: When the letter "س" exists in the prefix part of the word, it is not possible to decide if the letter "س" is a constant letter or not. For instance, the word" استماع".

c) Prefix letter ف

Initially, the letter {ف} is a non-constant letter. It can be converted to a constant letter by applying the following rules:

Rule1: If the letter" ف" exists after the first constant letter, the letter "ف " is treated as a constant letter. For example, with the words "أجحف", the letters "ح ,ج" have been identified constant letters. The letter "ف" exists after the first constant letter. Hence, the letter "ف" is treated as a constant letter. The constants' letters list becomes "ح ,ح ,ج".

Rule2: Check the position of the letter "ف" in the word. If the letter "ف" exists in the second half of the word, it is treated as a constant letter. For example, consider the word" استلف". The letter "ف" position in the second half of the word. Thus, in this case, it is considered a constant letter.

Rule3: If the letter "ف" is preceded by the letters "ال", it is treated as a constant letter. As it is in the word" الفنون".

Rule4: The letter "ف" is treated as a constant letter if it has been preceded by one of these letters "ت, ن, س, م, هـ, ي". As it is in the following words"يفلس, مفلس, سفيه, هفوف, نفلس, تفلس,"

d) Prefix letter ب

Initially, the letter {ب} is a non-constant letter. It can be converted to a constant letter by applying the following rules:

Rule1: -If the letter "ب" exists after the first constant letter, the letter "ب" is treated a constant letter. For example, with the word "صباح", the letters {ص, ح} have been identified constant letters. The letter "ب" exists after the first constant letter. So, the letter "ب" is treated as a constant letter. The constants letters' list becomes "ح, ب, ص".

Rule2: Check the position of the letter "ب" in the word. If the letter "ب" exists in the second half of the word, it is treated as a constant letter. For example, in the word "سالب", the letter "ب" positioned in the second half of the word. Thus, in this case, it is considered a constant letter.

Rule3: If the letter "ب" is preceded by the letters"ال", it is treated as a constant letter. As it is in the following word" الباسل".

Rule4: If the letter "ب" location is more than two in the word, it is treated as a constant letter. As it is in the word "الأبدين".

Rule5: The letter "ب" is treated as a constant letter if it has been preceded by one of these letters "ب, أ, ا, ت, ن, م, س, هـ, ي". As it is in the following words",أباركم, ابان, ببعض, سباق, هبوب, يبتلع نبدأ".

Rule6: When the letter "ب" exists in the prefix part of the word, It is not possible to decide if the letter "ب" is a constant letter or not. Such as the word" باسل".

2) Suffix letter "هـ": Suffix letter is one of the non-constant letters and attached at the end of the words. A certain set of rules has been implemented to convert this letter from non-constant letter to a constant letter. In this algorithm "هـ" is the only suffix letter.

The letter "هـ" is treated as a non-constant letter if the letter "هـ" exists in the suffix part of the word. The letter "هـ" is treated as an original root letter if it exists in places rather than the suffix part of the word. Initially, the letter "هـ" is a non-constant letter. It can be converted into a constant letter by applying the following rules:

Rule 1: If the letter "هـ" exists before the last constant letter, the letter "هـ" is treated as a constant letter. For example, in the word "اجتهد", the letters "ج, د" have been identified as a constant letter. The letter "هـ" exists before the last constant letter. So, "هـ" is treated as a constant letter. The constants letters list becomes "ج, هـ, د".

Rule 2: Check the position of the letter "هـ" in the word. If the letter "هـ" exists in the first half of the word, it is treated as a constant letter. For example, consider the word" تهامة". The letter "هـ" position is in the first half of the word. So, in this case, it is considered a constant letter.

Rule 3: The letter "هـ" is considered as a constant letter if the letters "وا" exist at the end of the word and the letter "هـ" appears just before the letters "وا" , such as " انتبهوا, أماهوا, تلاهوا".

Rule 4: The letter "هـ" is treated as a constant letter if it has been preceded by one of the letters "ب, س, ف, ك, ل, هـ, ة", such as" الدهس, التلهف, أشبهك, فقاهة, اسهله, تأهل, أسهب,"

3) The prefix-suffix letters "ك, ن, م": The Prefix-Suffix letters {ك, ن, م} are non-constant letters; a certain set of rules has been implemented on each letter on the Prefix-Suffix letters' list in order to convert these letters from non-constant letters to constant letters. The Prefix-Suffix letters are treated as constant letters to the root if these letters exist in the Prefix part or the suffix part or of the word. In contrast, they are treated as original root letters if they exist in the places rather than the prefix part or the suffix part of the word.

a) The Prefix-Suffix letter " ك"

Initially, the letter "ك" is a non-constant letter; it can be converted to a constant letter by applying the following rules:

Rule1: The letter " ك" is treated as an original root letter if it exists between constant letters. In the word "الشكر", the letters "ش, ر" are identified as a root letter. Then the letter "ك" exists between the two constants letters, "ك" letter is treated as a constant letter also. Thus, the constant letters list is "ر, ك, ش".

Rule2: The letter "ك" is considered as a constant letter if it appears in the first half of the word and not following the " ف, و" letters, such as " أكلائها, الكلمات, مكاسر".

Rule3: The letter "ك" is considered as a constant letter if it appears in the second half of the word and before the last constant letter, such as "النكاح, المنكر".

Rule4: The letter "ك" is considered as a constant letter if it appears in the second half of the word and has been followed by ", ات, ت, ين" letters, such as "تباكت, المساكين, المؤتفكات".

Rule5: When the letter "ك" exists in the prefix or suffix part of the word, it is not possible to decide if "ك" is a constant letter or not. The word is ambiguous, such as "كثيب, شراك"

b) The Prefix-Suffix letter " م"

Initially, the letter "م" is a non-constant letter; it can be converted to a constant letter by applying the following rules:

Rule1: The letter "م" is treated as an original root letter if it exists between constant letters. For example, with the word "أعماق", the letters "ع, ق" are identified as root letters, the letter "م" exists between the two constants letters, and "م" letter is treated as a constant letter. "م" letter is added to the constants letters list. In this case, one of the possible roots for the word "أعماق" is"عمق".

Rule2: The letter "م" is considered a constant letter if it appears in the second part of the word and before the last constant letter, such as "بالتمكث, والمكر".

Rule 3: The letter "م" is considered a constant letter if it appears in the first part of the word and is positioned after the first constant letter, such as "ثميلة" "

Rule4: The letter "م" is treated as a constant letter if it appears in the first part of the word and has been preceded by one of the letters "ي, ت, ا, such as "يمشي, امتنع ، تمدح".

Rule5: The letter "م" is considered a constant letter if it has been preceded by one of the letters "ا, و, ي" only in case the letters "ا, و, ي" are appeared after the last constant letter in the word, such as "الرحيم, طعام, القوم".

Rule6: The letter "م" is considered a constant letter if it appears just after the last constant in the word, such as " الرحمن, العجم".

Rule7: The letter "م" is considered a constant letter if it appears in the second part of the word and followed by the letter "ات", such as "السالمات".

Rule8: The letter "م" is considered a non-constant letter if the word consists of three constant letters and the letter "م" appears just before the first constant letter, such as " للمطبوع, المفضل".

c) The Prefix-Suffix letter "ن"

Initially, the letter "ن" is a non-constant letter; it can be converted to a constant letter by applying the following rules:

Rule1: The letter "ن" is treated as an original root letter if it exists between constant letters. For example, in the word "أعناق", the letters "ق ,ع" have been identified as constant letters. Then the letter "ن" is treated as a constant letter. The letter "ن" is added to the constant letters list. In this case, one of the possible roots for the word "أعناق" is the root "عنق".

Rule2: The letter "ن" is considered as a constant letter if it appears in the first part of the word and it has been preceded by one of these letters "ن ,ال" , such as "النواحي, ننشئهم".

Rule3: The letter "ن" is considered a constant letter if the word ended with the following letters "مين, ماء" , such as "الأيهمين, الأظماء".

Rule4: The letter "ن" is considered as a constant letter if it appears in the second part of the word and followed by the last constant letter, such as the words"المؤنث, بالاستنجاء" .

Rule5: The letter "ن" is considered a non-constant letter if the word consists of three constant letters and "ن" letter appears just before the first constant letter, such as " "وينزعج.

4) The uncertain letters "ء, و, أ, إ, ئ,ؤ": The uncertain letters "ء, و, أ, إ, ئ, ؤ" can appear in any part of the word. A certain set of rules has been implemented on each letter to convert these letters from non-constant letters to a constant letter by applying the following rules:

Rule1: The letter "ؤ" is considered as a constant letter if it not proceeded by one of these letters "ا, ي, و, أ", such as "المؤلف, البؤس".

Rule2: The letter "ئ" is considered as a constant letter if it appears in the word and has not been preceded by one of the letters "ا, ي, و" such as "الذئب, المئبر".

Rule3: The letter "ا" letter is considered as a constant letter if it appears in the word and followed by the letter "ة", such as "الصلاة, الرعاة".

Rule4: The letter "ء" is considered as a constant letter if it appears in the word and has not been preceded by one of the letters "ا, ي, و, أ", such as "المرء".

Rule5: The letter "و" is considered as a constant letter if it appears in the word and has been preceded by "ال" letter, such as "الولايات, الوليمه".

5) The extra letter "ة": The extra letter "ة" is not from the root's letter. Therefore, we remove this letter from the word.

D. Extracting All Possible Patterns for the Word

In the previous step, finding all constant letters will minimize the possible root's letters. The problem is when the algorithm does not find three constant letters or more, the algorithm tries to expert each letter in the word to complete the missing letter in the root.

1) Extracting all possible patterns when constant letters are three or more

Most of the Arabic words are derived from trilateral Arabic roots. However, there are very few quadric-literal Arabic roots relative to the number of trilateral Arabic roots. Most of the studies which related to Arabic Root extraction either are based on a dictionary of Arabic roots or use a set of rules to identify the verb patterns of the Arabic words. The rules are selected depending on the number of letters in the word to find the Arabic roots. In this section, we explain how to extract all possible patterns when constant letters are three or more. One possible verbal pattern exists if the word consists of three or more constant letters. The steps are summarized in Table II.

TABLE II. EXTRACTING PATTERNS WHEN CONSTANT LETTERS ARE THREE OR MORE

No.	Steps	Result
1	The input word	الحاشدون
2	Find constant letters	ح ش د
3	If constant letters are more than two letters, replace the first constant letter with "ف" letter; then replace the second constant letter with "ع" letter, after that replace the rest of constant letter with "ل" letter.	الفاعلون

According to Table II, with the word "الحاشدون," three constant letters are found in the word, which is forming one possible root for the word; it is "حشد." To find the pattern of the word "الحاشدون", replace "ح" letter with "ف" letter; then replace "ش" letter with "ع" letter, after that "د" letter with "ل" letter, in order to achieve the pattern"الفاعلون ". Another example that contains four constant letters with the word "المتدحرجات," . There are four constant letters forming the possible root "دحرج." Therefore, the pattern is "المتفعللات."

2) Extracting all possible patterns when constant letters are less than three

If the number of constant letters less than three letters, there will be more than one possible pattern. For instance, in the word"التقى," there is just only one constant letter "ق." In this case, we cannot build the pattern because we should have at least three constant letters to build the complete pattern. Therefore, the algorithm tries to find another two constant letters in the word in order to form the correct possible patterns. Note that all letters in the word are the candidate to be constant letters. Therefore, the suggested letters are "ال, ات, ای, لت, لی, تی". Referring to the suggested letters, the possible patterns are "الفعل, افتعل,افعلی,فلتعل, فلعلی,فعتلی" . Refer the process in Table III.

TABLE III. STEPS TO EXTRACT ALL POSSIBLE PATTERNS WITH JUST ONE CONSTANT LETTER

No.	Steps	Result
1	The input word	التقى
2	Find constant letters	ق
3	Find all possible other two letters	ال, ات, ای, لت, لی, تی
4	Extract all possible patterns	الفعل, افتعل,افعلی,فلتعل,فلعلی,فعتلی

3) Exclude the wrong patterns by applying the rules

In the previous section, some of the extracting of the possible patterns "الفعل, افتعل,افعلی,فلتعل,فلعلی,فعتلی ", patterns are wrong because the non-constant letters in the patterns are in the wrong places. So, the present algorithm applies the rules in section C in order to reject the wrong patterns.

TABLE IV. EXAMPLE OF EXCLUDING THE WRONG PATTERNS BY APPLYING THE RULES

No.	Steps	Result
1	All possible patterns	الفعل, افتعل,افعلی,فلتعل,فلعلی,فعتلی
2	Check the rules of non-constant letters	فلتعل,فلعلی are rejected
3	The accepted patterns	الفعل, افتعل,افعلی,فعتلی

With the word"التقى," the suggested patterns" فلتعل,فلعلی " are removed from the list after applying the rules. As indicated earlier for "ل" letter rule, if "ل" letter exists between two constant letters, it is considered as a constant letter. At the same time, it could not be extra letters. Therefore, the possible patterns are "الفعل , افعلی, افتعل,فعتلی" see the Table IV.

4) Minimizing the possible patterns by comparing them with patterns' list

A list of patterns in the corpus of Thalji [24] is automatically extracted and this list contains "4320" patterns. In order to ensure that the possible patterns are correct, they are compared with the patterns' list if they are not found they are rejected. For example, for the possible patterns in the word"التقى", the pattern "فعتلی" is not found in the list, so it is rejected, and the remaining possible patterns are"الفعل, افتعل,افعلی".

TABLE V. MINIMIZING PATTERNS BY COMPARING THEM WITH PATTERNS' LIST

Steps	Result
Possible patterns	الفعل, افتعل,افعلی
Compare the patterns with a pattern list	فعتلی is rejected
The accepted patterns	الفعل, افتعل,افعلی

E. Extract All Possible Roots for the Word

1) Finding all possible roots by matching the patterns

After finding all possible patterns, now all the possible roots that match the patterns are extracted. For example, in the word"التقى", the possible patterns are"الفعل, افتعل, افعلی" . Therefore, the possible roots are"تقی, لقی, لتق".

2) Finding all possible roots by applying Ebdal rules

After careful and considered review of the content of the Arabic dictionaries such as "Lessan AL-Arab" [25], it has been found that this dictionary has roots like"صطبل, صطفل, ضطر, زدر زدف". However, this the dictionary doesn't apply the Ebdal rule [26]. So, the Ebdal rules are always not applied. In our algorithm all the possible roots are returned with applying Ebdal rules and with don't apply it, to be in the safe side. Our proposed algorithm returns all the possible roots for each word by applying Ebdal rule and returning suggested roots without applying it.

F. Solve the Problem with Shaddah

This work is for the non-vocalized text. So, in many cases the writers don't write Shaddah above the letter, hence, the algorithm will try to check for missing Shaddah. It is started from the second letter in the word to check for missing Shaddah for all letters except the vowel letters. For example, the word "البر", the algorithm is generated by these possible missing Shaddah, "اللبر, الببر, البرر".

G. Solve the Problem with a Missing Vowel in Ealal Rules

In Arabic language, if the root has one or more long vowel, in derivation words these letters may be deleted. For example, for the root "قول", one of possible derivation word is "قل". During the derivation process, the long vowel "و" letter is deleted. So, in this case, the algorithm gives all possible cases of missing long vowel letter. The algorithm is generated these possible missing vowels "وقل, قول, قلو".

H. Solve the Problem by Changing the Vowel in Ealal Rules

In Arabic language, if the root has one or more long vowel, in derivation words these letters may be changed to deferent long vowel letter [27]. For example, with the root "قول", one of possible derivation word is "قال". During the derivation process, the long vowel "و" letter is changed to different long vowel letter, which is "ا". So, the algorithm gives all possible cases of changing long vowel letters. For example, with the word "قال", the algorithm is generated these possible different vowels "قيل, قول".

I. Minimizing Possible Roots by Comparing them with Roots' List

The algorithm generates a large number of possible roots, especially for words that found less than three constant letters and for vowel roots, because vowel roots have many more special cases. The presented algorithm uses the roots' list of Thalji [24] to minimize the possible roots. This list has 12000 roots. For example, the case of "درهم" word, the algorithm generates these root دير, درا, درو, دري, درر ادر, ودر, يدر, دار, دور, but the root "يدر " is excluded, because it is not founded in root's list.

J. Solve the Problem with Length One Words

In Arabic, there are some few words with only one letter length like "قِ، ع، ر", these words are derived from vowel root with length three letters, and these vowel letters are deleted during derivation process. The presented algorithm tries to find the root for such words by generating all possible vowels roots and all permutations. For example, with the word "قِ" all possible vowel letters are listed in Table VI, then these roots are compared with roots list of Thalji if these roots are founded the root is accepted otherwise the root is rejected.

TABLE VI. GENERATED ROOTS FOR WORD "قِ"

Generated root	Accepted or not
اقا	Accepted
اقو	Not accepted
اقي	Not accepted
ااق	Not accepted
اوق	Accepted
ايق	Accepted
وقا	Not accepted
وقو	Not accepted
وقي	Accepted
واق	Accepted
ووق	Accepted
ويق	Not accepted
يقا	Not accepted
يقو	Not accepted
يقي	Not accepted
ياق	Not accepted
يوق	Not accepted
ييق	Not accepted

K. Try to Find Other Roots

For words like the word "درهم", the algorithm result only includes these roots " درر, درأ، دري، دور، ودر ". In this word's case, the algorithm just finds two consonant letter and tries to find the third one, not the fourth one also. So, it misses the root "درهم" . In this case, the algorithm tries to check the word itself "درهم", since it's length is four. So, the result is درهم, درأ، درر ، دري، دور، ودر".

IV. Experiment and Evaluation

In this section, the presented algorithm is compared with the Arabic root extraction algorithm of Khoja and Garside, which is the most popular Arabic root extraction algorithm, and the only Arabic root extraction algorithm that publicly available for download. Khoja and Garside tested their Arabic root extraction algorithm using newspaper text and achieved 95%. Specifically, we make a pure and completely comparison between the algorithm of Khoja and Garside and the presented algorithm on the corpus of Thalji. Thalji's corpus is an automatic corpus that is built from ten old Arabic dictionaries. This corpus is mainly built to test and fairly compare Arabic roots extraction algorithms. This corpus contains 720,000 words roots pair, which helps to avoid the interference of a human expert normally needed to verify the correct roots of each word used in the testing or comparison process. Moreover, this corpus has more than 4,320 types of words which derived from (12000) roots. So, it guarantees the comprehensiveness of words.

The experimental result shows that the accuracy of the algorithm of Khoja and Garside is 63%, and the accuracy of the presented algorithm achieves 94%. As shown in Figure I.

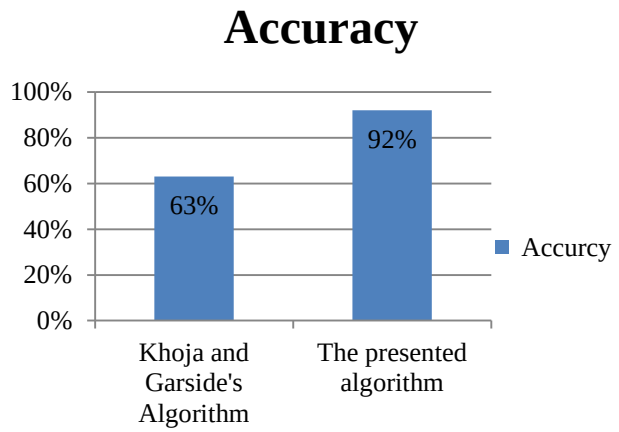

Fig. 1. Accuracy of Khoja Algorithm and the Presented Algorithm.

We observed that the following limitations caused the decrement of accuracy for the algorithm of Khoja and Garside:

1) The algorithm of Khoja and Garside is missing a large number of roots, prefixes, suffixes, and patterns. The dictionary of Khoja and Garside is restricting the result for just 4,748 roots, 3,822 trilateral roots, 926 quadrilateral roots. Because the algorithm of Khoja and Garside ignores 7252 roots, the result of ignoring these roots causes wrong results because if one uses any of ignoring roots, he/ she will not find

the correct result or will not achieve the correct root. For example, the word "مملوع" is stemmed to the wrong root "ولع", because it misses the root "ملع", also the word "لأصدقائك" is not stemmed because it is missing the pattern "لأفعلائك", the same thing will occur to the following words: "اخلعوهن, يحرمونهن", "تستبدل, حاسوب, ازدهر, اصطحب, اصطلح, لأصدقائك, نستخدمها, شركاؤنا".

2) The algorithm of Khoja and Garside suffers from affix ambiguity problems. For example, it returns "ميع" root for the word "استماع", but it should also return the root "سمع", this is because it starts by removing the longest suffix or prefix, but sometimes its neither prefix nor suffix, its root's letters.

3) The algorithm of Khoja and Garside, again, returns just one solution for non-vocalized words, ignoring other possible solutions. For example, the word "قل", the possible roots are "قول, قلل, قلو, وقل, قيل قلي", where the result is just "قلل".

4) The algorithm of Khoja and Garside replaces a weak letter with the letter "و", which occasionally produces a root that is not related to the original word. For example, it returns "صول" root for the word "أصيل" which is the wrong root, the right root is "أصل".

5) The algorithm of Khoja and Garside may generate invalid roots or fail to find roots for words that contain Ebdal rule "ابدال" like "اصطلح, اصطحب" and "ازدهر".

6) The algorithm of Khoja and Garside, also, doesn't deal with Shaddah. For example, with the word "وأب", it returns the root "أبي", where the possible root also is "أبب".

To be fair, the followings are the limitation points of the proposed algorithm:

7) The presented algorithm unable to find the root of words is in the word "نيعوعة", the algorithm result is just "دعع" root. In this well-known word's case, if the algorithm finds three constant letters, it returns them as trilateral root which becomes the result. At the same time, the presented algorithm is not deal with exchanging the constant letter with the vowel letter because this case rarely happened.

8) The presented algorithm gives all possible roots of the word. However, this causes a misunderstanding result for the researcher to find which the exact root for the word is. This limitation coming up clearly because the presented algorithm deals with words rather than completed meaningful sentences in a paragraph.

V. Conclusion and Future Work

In this study, we investigate the rules which are based on the existing Arabic root extraction, analyse most previous Arabic root extraction algorithms, inspired by all their strong ideas, and overcome the weaknesses' points. This study continues what the others already started by performing extensive enhancement and improvements.

The presented Arabic root extraction algorithm is compared with the Arabic root extraction algorithm of Khoja and Garside, which is a well-known Arabic root extraction algorithm. The algorithm of Khoja and Garside yields 95% of accuracy when it was tested in the selective data set. However, the experimental result shows 63% accuracy when we tested their algorithm using Thalji's corpus. At the same time, we test the proposed algorithm on the same corpus and able to achieve 94%. The main reason of decreasing the percentage of the algorithm of Khoja and Garside from 95% to 63% is because of the different datasets that are used in the testing process. This proved that the algorithm of Khoja and Garside has insufficient rules to handle bigger test data with wider diversity and variation of words.

We plan to enhance the accuracy of the presented algorithm by solving its weakness points as stated in the above. The future works will include the enhancement of rules to obtain just the exact root instead of multiple roots, which requires the algorithm to analyse and understand the sentence or sometimes the paragraph.

References

[1] G. Kanaan, R. Al-shalabi and M. Sawalha, "Improving Arabic information retrieval systems using part of speech tagging," pp. 32-37, 2005.

[2] Z. Thalji, "A New Algorithm to Minimize Names in the Arabic Language," International Journal of Applied Engineering Research, vol. 13, no. 18, pp. 13950-13960, 2018.

[3] S. Al hakeem, G. Shakah, B. Abu Saleh and N. Thalji, "Developing an effective light stemmer for Arabic language information retrieval," International Journal of Computer and Information Technology, vol. 5, no. 1, pp. 55-59, 2016.

[4] T. M. T. Sembok and B. AbuAta, "Arabic word stemming algorithms and retrieval effectiveness," In Proceedings of the World Congress on Engineering , vol. 3, pp. 3-5, 2013.

[5] R. Kanaan and G. Kanaan, "An improved algorithm for the extraction of triliteral Arabic roots," European Scientific Journal, vol. 10, no. 3, pp. 346-355, 2014.

[6] B. Abuata and A. Al-Omari, "A rule-based stemmer for Arabic Gulf dialect," Journal of King Saud University-Computer and Information Sciences, vol. 27, no. 2, pp. 104-112., 2015.

[7] E. Al-shawakfa, A. Al-Badarneh, S. Shatnawi, K. Al-Rabab'ah and B. Bani-Ismail, "A Comparison study of some Arabic root finding," Journal Of The American Society For Information Science And Technology, vol. 61, no. 5, pp. 1015-1024, 2010.

[8] S. Al-Fedaghi and F. S. Al-Anzi, "A new algorithm to generate Arabic root-pattern forms," proceedings of the 11th national Computer Conference and Exhibition, 1989.

[9] R. Al-shalabi and M. Evens, "A Computational Morphology System for Arabic," In Proceedings of the Workshop on Computational Approaches to Semitic Languages. Association for Computational Linguistics., pp. 66-72, 1998.

[10] H. M. Al-Serhan, R. Al Shalabi and G. Kannan, "New approach for extracting Arabic roots," Proceedings of the 2003 Arab conference on Information Technology, pp. 42-59, 2003.

[11] S. Khoja and R. Garside, "Stemming Arabic text," Lancaster, UK, Computing Department, Lancaster University, 1999.

[12] R. Alshalabi, "Pattern-Based Stemmer for Finding Arabic Roots," Information Technology Journal, pp. 38-43., 2005.

[13] M. N. Al-kabi and R. AL-Mustafa, "Arabic Root Based Stemmer," Proceedings of the International Arab Conference on Information Technology, 2006.

[14] R. Sonbol, N. Ghneim and M. S. Desouki, "Arabic Morphological Analysis : a New Approach," In Information and Communication Technologies: From Theory to Applications, 3rd International Conference, IEEE, pp. 1-6, 2008.

[15] S. Ghwanmeh, S. Rabab'Ah, R. Al-Shalabi and G. Kanaan, "Enhanced Algorithm for Extracting the Root of Arabic Words," Sixth International Conference on Computer Graphics, Imaging and Visualization, pp. 388-391, 2009.

[16] Z. Kchaou and S. Kanoun, "Arabic stemming with two dictionaries," In Innovations in Information Technology, International Conference IEEE., pp. 688-691, 2008.

[17] M. El-Defrawy, Y. El-Sonbaty and N. Belal, "A Rule-Based Subject-Correlated Arabic Stemmer," Arabian Journal for Science and Engineering , vol. 41, no. 8, pp. 2883-2891, 2016.

[18] A. Ayedh and T. Guanzheng, "Building and Benchmarking Novel Arabic Stemmer for Document Classification," Journal of Computational and Theoretical Nanoscience, vol. 13, no. 3, pp. 1527-1535, 2016.

[19] K. Taghva, R. Elkhoury and J. Coombs, "Arabic Stemming without a root dictionary," In Information Technology: Coding and Computing, International Conference, IEEE, pp. 152-157, 2005.

[20] E. Al-Shammari and J. Lin, "A novel Arabic lemmatization algorithm," Al-Shammari, E., & Lin, J. (2008, JulyIn Proceedings of the second workshop on Analytics for noisy unstructured text data ACM., pp. 113-118, 2008.

[21] M. N. Al-Kabi, S. A. Kazakzeh, B. M. Abu Ata, S. A. Al-Rababah and I. M. Alsmadi, "A novel root based Arabic stemmer," Journal of King Saud University-Computer and Information Sciences, pp. 94-103, 2015.

[22] A.-K. N. Al-Kabi, "Towards Improving Khoja Rule-Based Arabic Stemmer," In Applied Electrical Engineering and Computing Technologies (AEECT), IEEE Jordan Conference, pp. 1-6, 2013.

[23] F. Abu Hawas and K. Emmert E, "Rule-based approach for Arabic root extraction: new rules to directly extract roots of Arabic words," Abu Hawas, F., & Emmert, K. E. (2014). Rule-based approach for Arabic rJournal of Computing and Information Technology, vol. 22, no. 1, pp. 57-68, 2014.

[24] N. Thalji, N. A. Hanin, Y. Yacob and S. Al-Hakeem, "Corpus for Test , Compare and Enhance Arabic Root Extraction Algorithms," International Journal of Advanced Computer Science and Applications, vol. 8, no. 5, pp. 229-236, 2017.

[25] M. Ibn Manzur, Lisan Al-Arab, no date.

[26] A. Abu altaeeb, Ebdal Book, Damascus: Arabic Language Group, 1961.

[27] E. Abdulaleem, Tayseer Ealal and Ebdal, Cairo: Dar Ghraib For Printing, Publishing and Distribution, 1993.