Computers, Materials & Continua
DOI:10.32604/cmc.2020.xxxxxx
Type: Article

Tech Science Press

A New Enhanced Arabic Light Stemmer for IR in Medical Documents

Ra’ed M. Al-Khatib[1,*], Taha Zerrouki[2], Mohammed Abu Shquier[3], Amar Balla[4] and Asef Al-Khateeb[5]

[1]Department of Computer Sciences, Yarmouk University, Irbid-21163, Jordan
[2]Bouira University, Faculty of sciences and applied Sciences, Bouira, Algeria
[3]Faculty of Computer Science and Information Technology, Jerash University, Jordan
[4]Ecole nationale Supérieure d’Informatique (ESI), Algiers, Algeria
[5]Imam Mohammad Ibn Saud Islamic University, College of Shari'a and Islamic Studies in Al Ahsaa, Department of Computer Science, Saudi Arabia

[*]Corresponding Author: Ra’ed M. Al-Khatib. Email: raed.m.alkhatib@yu.edu.jo
Received: 25 December 2020; Accepted: XX Month 202X

Abstract: This paper introduces a new enhanced Arabic stemming algorithm for solving the information retrieval problem, especially in medical documents. Our proposed algorithm is a light stemming algorithm for extracting stems and roots from the input data. One of the main challenges facing the light stemming algorithm is cutting off the input word, to extract the initial segments. When initiating the light stemmer with strong initial segments, the final extracting stems and roots will be more accurate. Therefore, a new enhanced segmentation based on deploying the Direct Acyclic Graph (DAG) model is utilized. In addition to extracting the powerful initial segments, the main two procedures (i.e., stems and roots extraction), should be also reinforced with more efficient operators to improve the final outputs. To validate the proposed enhanced stemmer, four data sets are used. The achieved stems and roots resulted from our proposed light stemmer are compared with the results obtained from five other well-known Arabic light stemmers using the same data sets. This evaluation process proved that the proposed enhanced stemmer outperformed other comparative stemmers.

Keywords: Machine learning; Information Retrieval (IR) systems; medical documents; stemming algorithms; Arabic light stemmer; Natural Language Processing (NLP)

1 Introduction

Stemming text is important in Natural Language Processing (NLP) domain since it is used in Information Retrieval (IR), spell checking, morphology analysis, text classification and clustering [1-3]. Arabic is a non-agglutinative language in derivational morphology level, which makes root extraction from a word more difficult. The main problem here is the ambiguity to find exact stem, or extract the exact root from derived word. For example: the input word “والدين” when it's not vocalized can lead into two different stemming extraction as “ϕ+والد+ين” (means the two parents), or “والـ+دين+ϕ” (means with the religion). Further, in some cases, the original stem is not extracted properly, for example “يستعمل” is extracted to be “يـ+ستعمل+ϕ” with wrong stem “ستعمل”, while the correct lemma/stem must be “استعمل”. Therefore, the main attention of computational studies in the Arabic NLP and morphological IR domain is to develop a proper and more efficient light stemming algorithm [4]. This is to tackle these difficulties in extracting the best and most accurate stem/root from the input Arabic words.

In this paper, we proposed a new enhanced stemming algorithm, which is interacting as light stemmer and segmentor algorithm at the same time. This proposed enhanced algorithm primarily supports light stemming (removing prefixes and suffixes) and extracts all possible segmentations from the input word. This stemmer is a new enhanced based on Directed Acyclic Graph (DAG) model adapted from [5], which empowered the proposed algorithm to generate all segmentations. Therefore, the enhanced algorithm offers stemming extraction and root extraction at the same time, unlike other existing Arabic light stemmers in literature; Khoja [6], ISRI [7], FARASA [8], and Assem [9]. This enhanced stemmer is also provided with default lists of prefixes and suffixes that can be customized and updated separately.

Consequently, the main contributions of our proposed enhanced stemmer are summarized as follows. (i) Accepting the use of customized prefixes and suffixes lists, which allow the end-user to handle and update these lists, and make customized stemmers without any changing to the main source code. (ii) The algorithm of this enhanced stemmer can be added to the Python library, and will also be available as a demo and as open-source code on Github. (iii) The proposed stemming algorithm model can be combined for building new IR and NLP systems.

The remaining parts of this paper are organized as follows. The morphological structures and word types are defined and overviewed in the next Section. In Section 3, we presented the main state-of-the-art Arabic stemmers existing in the literature. Our proposed enhanced light stemming algorithm is fully discussed in Section 4. The comprehensive experiments and comparative evaluations are introduced in Section 4, and finally, we conclude this work with future directions in the last section.

2 Research Background

In order to make this research paper a self-exploratory, the various structures of the Arabic word and the parts of speech in Arabic language will be defined and distinguished in this section. In Arabic language, morphology or the word structure domain is primarily considered the consonant root to obtain the final forms of many derived words [10, 11]. IR systems mostly used stem, pattern, and lemma in addition to the root. Thus, the main differences among root, stem, pattern, and lemma are obtained from the way that they work. Therefore, the results and definitions of these structural categories can be returned as follows:

Root: In Arabic language, the root is an invariable discontinuous bound morpheme that can be represented by two to five phonemes. Basically, three consonant letters in certain order interlocks with the pattern to form a stem, where the root has lexical meaning [10]. Aronoff defines the root in more vivid terms: "*A root is what is left where all morphological structure has been wrung out of a form. This is the sense of the term in Semitic grammar*" [12].

Stem or word stem: Stem is the base or bare form of the word without inflectional affixes. Therefore, the lexeme may have more than one stem [13].

Pattern: The pattern is a discontinuous morpheme consisting of one or more vowels and slots for root-phonemes (means radicals). The pattern comes either alone or in combination with one to three derivational affixes, which interlocks with a root to format a stem. So, the stem has grammatical meaning in general [13].

Stemming: Stemming process in Arabic is performed by cutting off the end and/or the beginning of the word, taking into account a list of common prefixes and suffixes that can be found in an inflected word. This indiscriminate cutting can be successful on some occasions, but not always at all.

Lemmatization: On the other hand, lemmatizing process takes into consideration the morphological analysis of the words. To do so, it is necessary to have detailed dictionaries that the algorithm can look through, to link and match the form back to its lemma.

In the summary, the word "المكتوبة" (means the written thing), is originally derived from the three consonant root "كتب". Further, the stem is "مكتوب" after removing the definite article "الـ" "al altaerif" as a prefix, and the feminine tool "ـة" as a suffix for singular feminine noun [14]. Therefore, the stem "مكتوب" matching with the pattern "مفعول", which can be used to extract the root "كتب". Here in this example

"المكتوبة" the lemma and stem are the same, which is "مكتوب". To distinguish between stem and lemma, let's take the derived word "يستكتبون" as another example for the input Arabic word. After removing the prefix "يـ" and the suffix "ـون", the stem will be "ستكتب". Then, the lemma is "استكتب" after adding the letter "alef" "ا" at the beginning. Therefore, in this paper, a novel light stemming algorithm based on the directed acyclic graph (DAG) model is proposed [5]. Then, this adapted powerful DAG model is firstly employed to extract all possible segments from the input word. These segments as stem-hits will be initial stems and lemmas for our proposed enhanced stemmer. Finally, the main operator of the proposed enhanced stemming algorithm will use these segments in the extraction process to provide the final outputs and to extract the best roots and stems.

3 Literature Review

Several stemming algorithms have been introduced in the literature to extract root, stem, and lemma from the input Arabic words. In [6], the researchers presented a light stemming algorithm named Khoja stemmer, which is the most popular Arabic light stemmer. Khoja stemmer removes prefixes and suffixes from the input word, often after performing a normalization process. Then, it matches the resulting word/segment to a pattern, to extracts the initial root. This initial root is validated against a list of correct Arabic roots to ensure linguistic correctness, and then to extract the final root. Khoja stemmer resolves the ambiguity by defining a set of linguistic rules based on some features such as the word's first character, prefixes, or suffixes length. Additionally, decisions are implicitly ordered whereas the final result will be the first correct root. Khoja stemmer also handles roots with duplicate letters first [15].

Taghva et al. [7] introduced the Information Science Research Institute's that well-known ISRI-stemmer. ISRI Arabic stemmer shared many features with the Khoja stemmer without using the root dictionary. Besides, if a root is not found, the ISRI stemmer returned normalized form, rather than returning the original unmodified word. Additional adjustments were made to improve the final results of the ISRI algorithm by adding new more 60 stop words and adding the pattern "تفاعيل" to the pattern set. ISRI light stemmer is available as part of the NLTK framework [16].

Abdelali et al. [8] developed FARASA stemmer, which is considered as text processing toolkit for Arabic text. FARASA consists of the segmentation/tokenization module, POS tagger, Arabic text diacritizer, and dependency parser. FARASA segmentation/tokenization module is based on SVM-rank with linear kernels, which uses a variety of features and lexicons to rank all possible segmentations of the word. These features include: (i) likelihoods of stems, prefixes, suffixes, their combinations; (ii) presence in lexicons containing valid stems or named entities; (iii) and underlying stem templates [8]. FARASA is also available for a demo with source code on GitHub.

Chelli [9] presented Assem stemmer as a light stemming algorithm, which is made based on the Snowball framework language [17]. Assem stemmer is fast and can be generated in many programming languages (through Snowball) [9]. Snowball is a small string processing language designed for creating stemming algorithms to be used in IR systems. Assem stemmer offers light stemming and text normalization. It can be configured to run as root extractor or stemmer, but in two separate packages, because the Snowball framework does not support stemming and rooting at the same time. The source code of Assem stemmer is also available on GitHub.

We discussed those Arabic stemmers listed in this section, in order to be used in the comparative and evaluation process as benchmarks against our proposed enhanced stemmer However, more several algorithms have been presented in the literature as Arabic light stemmers, which are categorized and illustrated in [18, 19]. Nevertheless, we cannot compare with them due to the lack of their source code, which is not available.

4 Proposed Stemming Algorithm

In this section, the proposed enhanced Arabic light stemming algorithm is discussed in detail. The key focus of this research paper is on the way of extracting all possible segmentations from the input Arabic word. Primarily, this segmentation process is performed based on deploying a directed acyclic graph (DAG) model that is discussed in the following subsection. The essential idea behind deploying DAG model is to provide strong initial segments for the main operator of our proposed enhanced stemming algorithm. This is to reinforce the final efficiency of the new enhanced stemming algorithm for extracting the best final roots and stems from the input data. The flowchart and general framework of the proposed stemming algorithm are illustrated in Fig. 1.

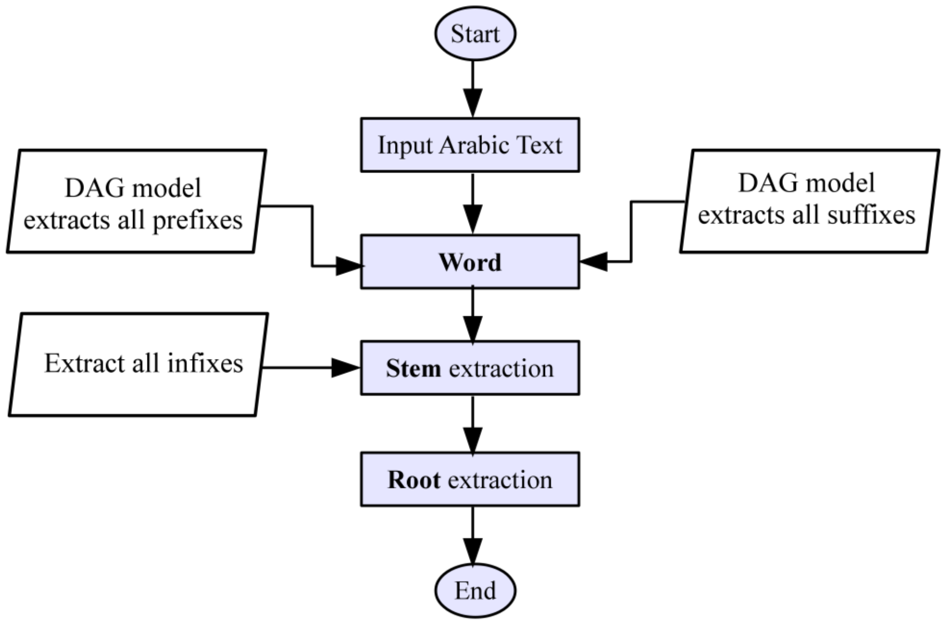

Figure 1: Flowchart of proposed enhanced Arabic light stemming algorithm

The proposed enhanced stemmer is initiated by a set of segments that are produced by operating the adapted DAG model [20], which can be run on-line in linear time with a strong correct sense. These segments as initial stem-hits are extracted after removing prefixes and suffixes via using the DAG model. The stem extraction operator is firstly used to extract final stems from these initial segments by running an indexing model, which is efficiently performed by translating numerical information to string. Secondly, there is a root extraction procedure that removes infixes from the input stems, to extract final roots. Finally, the proposed enhanced stemming algorithm is terminated after choosing the shortest stem, to extract the best final root. In a nutshell, our proposed enhanced light stemmer achieved the best outcomes of final stems and roots with reasonable complexity. The subsequent sections will thoroughly discuss each step of our proposed Arabic light stemming algorithm.

4.1 Extracting Initial Segments by DAG Model

For language stemming problem, the input Arabic word in the proposed algorithm can be represented as $w = (l_1, l_2, \ldots, l_N)$ of N letters. Then, the value range of each input Arabic word can be indexed by w_i, where w_i is the word labeled $w_i \in (1, 2, \ldots, N)$. The focal point is how to extract the initial segments from the input word w_i. The initial segments are primarily extracted based on the DAG model using nested tree-inspired and customized from [21]. As conventionally known, the adapted directed acyclic graph extracts affixes (prefixes and suffixes), using predefined affixes lists. Therefore, the DAG model extracts all possible affixation from the input Arabic word and provides all possible segmentations. The main feature of the proposed enhanced stemmer is data-independent, which does not perform a specific treatment according to a word list. This feature makes the enhanced proposed stemming algorithm fast to reduce data entry and validation. Consequently, our enhanced stemmer has a direct impact to be easily used in information retrieval systems, fast text analysis, and classification.

4.2 Prefix Extraction

For prefix extraction, the proposed enhanced stemming algorithm used a modified DAG model by taking a prefixes list, to give a non-deterministic structure for all possible solutions like a tree. For example, let assume the prefixes list has following {و، والـ، والمـ، وكـ، وكـالـ، وكـالمـ، وبـ، وبالـ، وبالمـ}. Then, the adapted DAG model resulted in prefixes nested tree as shown in Fig. 2, for input words { والمكتب، وكـالمكتب، وبالمكتب}. Finally, this nested tree will stop if there are no more choices of prefixes in the predefined list. During the process of DAG model, the automaton of a nested tree will save the letter positions at each time finds a finite state for each predefined prefix. Each letter position is the last letter for each prefix in the predefined list that is numbered in Fig. 2 and is represented using a double circle in the nested tree.

To analyze the extraction process for letter positions in Fig. 2, the extracted position numbers from right = {1,2,4,5}, will determine a list of ending position of prefixes for the input word “وبالمكتب”. This means all possible prefixes are: { و، وبـ، وبالـ، وبالمـ } (see right bottom example in Fig. 2. To decide the termination case, the DAG model chooses the longest prefix, which is “وبالمـ” as a default prefix. This will extract the “كتب” as an initial shortest segment.

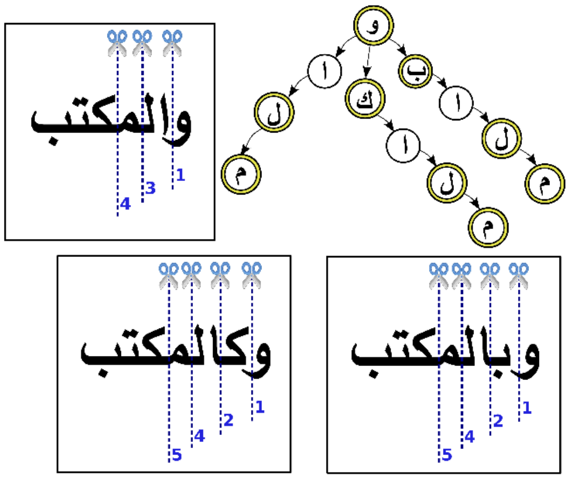

Figure 2: Prefixes extraction example based on the DAG model using nested tree

4.3 Suffix Extraction

The suffixes extraction process is performed in an inverse way, which is starting from the ending of the input word. The DAG model runs in inverse to search for all possible paths, as seen in Fig. 3. Then, it stops when there is no issue, and saves all positions of finite states. These saved positions will be labeled each start position of suffixes in the input word. For example, if the following: { ا، ها، ينها، وها، ونها }, are in the predefined list of suffixes. Then, the extracted position numbers for the input word “يكتبونها”, will be from right = {5,7,8}, as seen in Fig. 3. These indexed numbers from right = {5,7,8}, will determine the list of ending suffix positions in the input word “يكتبونها”. See the left top example in Fig. 3, to decide the termination case with mapping to the finite states of the nested tree. DAG model will also choose the longest suffix, which is “ـونها” as a default suffix to be removed. This will extract the “يكتبـ” as an initial shortest segment from the input word “يكتبونها”. On the other hand, the prefix extraction process in the previous step will also remove the letter “يـ” from the beginning of “يـ+كتبـ”, and finally extract “كتب” as the main shortest stem.

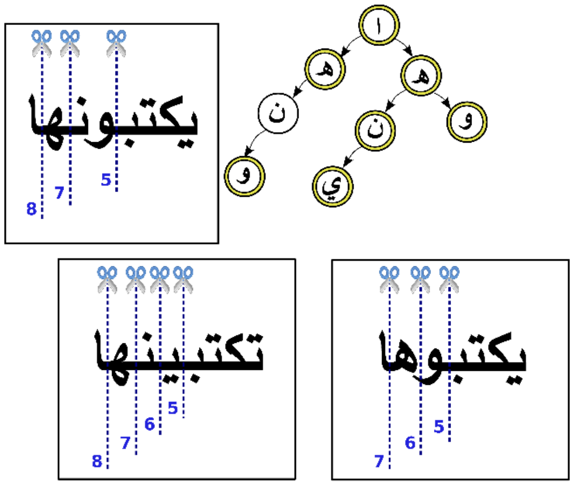

Figure 3: Suffixes extraction example based on the DAG model in an inverse way

4.4 Stem Extraction

Initially, the algorithm of stems extraction in our proposed enhanced stemmer will segment and cut off the input word into all possible segments. To get one stem, the longest prefix and longest suffix are removed, where the remaining part of the segment is the final shortest stem. In other words, the default stem will be the shortest segment from all possibilities of extracted segments.

In this step, the stem extraction process is terminated after one-to-one verification of prefixes and suffixes' compatibilities. After that, it removes all incompatible affixes (the incompatible prefixes and suffixes). Therefore, the final stem will be the shortest one from all accepted segments. In the following is a complete example to show how this stems-extraction procedure in the proposed enhanced stemmer will segments the input word "فسيكتبونهما". The stem extraction process will generate 20 possibilities of segments, which can be shown in their positions in tuples as follows.

[1]: (7 , 2)	[2]: (8 , 3),	[3]: (8 , 0),	[4]: (8 , 2),
فس+يكتبو+نهما	فسي+كتبون+هما	ϕ+فسيكتبون+هما	فس+يكتبون+هما
[5]: (10 , 3),	[6]: (11 , 2),	[7]: (8 , 1),	[8]: (7 , 0),
فسي+كتبونهم+ا	فس+يكتبونهما+ϕ	ف+سيكتبون+هما	ϕ+فسيكتبو+نهما
[9]: (10 , 2),	[10]: (11 , 3),	[11]: (6 , 0),	[12]: (11 , 0),
فس+يكتبونهم+ا	فسي+كتبونهما+ϕ	ϕ+فسيكتب+ونهما	ϕ+فسيكتبونهما+ϕ
[13]: (6 , 2),	[14]: (6 , 1),	[15]: (10 , 0),	[16]: (6 , 3),
فس+يكتب+ونهما	ف+سيكتب+ونهما	ϕ+فسيكتبونهم+ا	فسي+كتب+ونهما
[17]: (7 , 1),	[18]: (10 , 1),	[19]: (7 , 3),	[20]: (11, 1)
ف+سيكتبو+نهما	ف+سيكتبونهم+ا	فسي+كتبو+نهما	ف+سيكتبونهما+ϕ

Among these 20 possibilities, the shortest stem is "كتب", which is in possible number 16 after removing the prefix "فسيـ", and the suffix "ـونهما". This feature of extracting the shortest final stem is implemented and added to our proposed enhanced stemmer, as a super layer after the stemming process by giving a list of accepted affixes tuples. For example to the accepted tuples, the prefix "الـ" is coming compatible with suffix "ـات", in noun word "الكتابات". However, the suffix "ـهم" will not come with the prefix "الـ" as accepted tuple, like the input word "كتاباتهم" with suffix "ـهم", but without prefix "الـ". As intermediate results, all acceptable segments from this step will be the main inputs to the next step of our proposed enhanced Arabic light stemmer.

4.5 Root Extraction

After the stemming step, our proposed enhanced stemmer will extract many roots from the input Arabic word using all stems that are extracted in the previous step. Here, the roots extraction procedure considers infix-letters, which are the internal letters used to derivate words or irregular plural or conjugation. Primarily, the letters { ا ، و ، ي ، ن ، ت ، ط ، د } as a predefined list of infixes, will be removed from the input stems to extract the roots. Fig. 4 illustrates two examples for root extraction process from two input derived words "بالكاتبة" and "بالعاملين". For complete examples, applying a root-extraction operator to the input stems { عامل، عمال، اعتمل } can lead to one root "عمل" after removing any infix-letter that are mapping to the same list of predefined infixes.

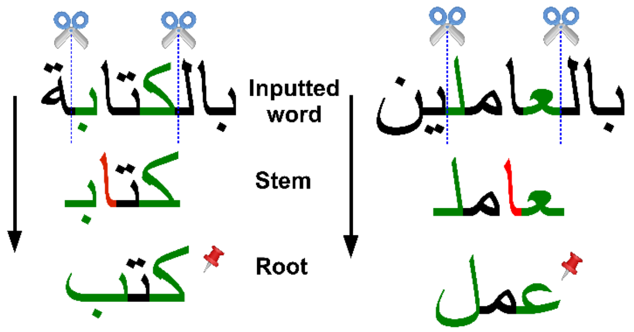

Figure 4: Root extraction based on removing any of {ا ، و ، ي ، ن ، ت ، ط ، د} as a predefined infixes

However, the root extraction process is not always easy and straightforward. Sometimes, the three infix letters { ت ، ط ، د } from this predefined list of infixes, will not always be removed from the input stem. These three letters { ت ، ط ، د }, are only removed in the following cases:

- The infix letter "ت", is removed if it comes after the first letter in the input stem, like stem "فتعل" from the input word "مفتعل" → ("مـ+فتعل" → "فتعل"). However, the infix letter "ت" will not remove from the input stem "كاتب", because the "ت" is the third letter in the stem, not the second letter, as seen in Fig. 4.
- The infix letter "ط", is removed if it comes after the letter "ض", in the input stem like "اضطرب".
- The infix letter "د", is removed if it comes after the letter "ز", in the input stem like "ازدهر".

Consequently, the final termination process of root-extraction for choosing the best final root will firstly ignore any root less than three letters. Then, it will choose the most frequent root among the rest extracted roots.

4.6 Customizing Proposed Stemmer

One of the important features of adapting this proposed enhanced stemming algorithm is the ability to customize this enhanced stemmer to build new ones without changing its original source code. The enhanced stemming algorithm can accept to handle any new affixes lists, and rebuild the automaton nested tree of DAG model in any new future stemmer or in any other IR system. This feature is the most important one for our enhanced stemmer, which allows the developers and researchers to customize affixes and make a specific domain or new light stemmer. Further, you can change the aim of the stemmer to extract roots by giving new derivational affixes, or extract stem [22, 23]. This is through the ability to build rules for removing syntactic affixes by giving a new syntactic affixes list [24].

In the following, a full example explains how we can deploy our enhanced stemmer as syntactic affixes removal by taking the derived word "بالمدرستين" as the main input. The existing algorithm of proposed enhanced stemmer will segment this input word "بالمدرستين" as follows: "بالمدرستين" → "بالـ+مدرست+ـين". This is done after removing the prefix "بالـ" from the beginning, and removing the suffix

"ـين" from the end. Here, the extracted stem will be "مدرست". Therefore, if the information retrieval application needs to use our enhanced stemmer to extract roots, the developer can customize the affixes list, to extract "درس" as the main stem from the input word "بالمدرستين" as follows: "بالمدرستين" → "بالمـ+ـدرس+ـتين". This will be done by removing the prefix "بالمـ" from the beginning, and removing the suffix "ـتين" from the end.

This can be handled using new affixes lists. In the following section, we exhibit the used lists of prefixes and suffixes:

- Prefixes: { بالمـ ، أ, أس, أف, أفس, أو, أوس, أول, س, ف, فس, فل, ل, و, وس, ول }.
- Suffixes: { تين ، ك, كم, كما, كن, نا, ني, هـ, ها, هم, هما, هن, ي }.

This feature will allow researchers to easily adapt our enhanced stemming algorithm with this step of customization in the Arabic NLP domain, and to use our proposed stemming algorithm in many information retrieval systems. In a nutshell, our proposed enhanced stemmer can be used in multiple contexts without changing the original source code of the main algorithm.

5 Evaluation Results and Discussion

In this section, a comprehensive discussion is presented for the experimental evaluation of outputs resulted from our enhanced Arabic light stemmer. Further, a comparison process is performed to show the prominence of our proposed enhanced stemmer that proved its superiority against the obtained results from other existing stemming algorithms in the literature. Basically, the used various Arabic data-sets are fully explained in the subsequent section, before discussing and analyzing the experimental evaluation results. To describe and validate the evaluation of the proposed stemmer, the outputs obtained from our proposed enhanced light stemmer are compared against the best outputs resulted from other five well-known light stemmers, which are Khoja [6], ISRI [7], FARASA [8], Assem [9], and Tashaphyne base stemmer [25]. Meanwhile, the main multiple variants of our proposed enhanced Arabic stemmer are: i) using customized prefixes and suffixes lists. ii) removing stop words. iii) using a preprocessing word tagging.

5.1 Used Data Sets

The used data sets for testing and evaluation of our enhanced stemmer, are carefully selected and revised from four different well-known Arabic benchmarks. These used four data sets are called: Gold Arabic corpus [26], NAFIS [27], Quranic Arabic corpus [28], and Quran index corpus [29]. Meanwhile, these used data sets are different in terms of word count, lemma/stem counts, and root counts. Tab. 1 summarizes the statistics of all these data sets.

In order to clarify the main features of each data set, a comprehensive description of the used four data sets can be explained as follows:

1) *Gold Arabic Corpus*: is initially built for testing Assem stemmer, and other Arabic stemmers [26]. The gold corpus can be freely downloaded.
2) *NAFIS Arabic Corpus*: Normalized Arabic Fragments for Inestimable Stemming (NAFIS) is an Arabic stemming standard corpus, which is composed of a collection of texts [27]. NAFIS is selected to be representative of Arabic stemming tasks, and they are manually annotated and processed.
3) *Quranic Arabic corpus*: is annotated linguistic corpus [28], which is a resource of the Arabic grammar, syntax, and morphology for each word in the Holy Quran [30]. This Quranic corpus provides three levels of analysis: the morphological annotation, the syntactic treebank, and the semantic ontology. This Quranic Arabic corpus differs from other Arabic banks of data in the following three important ways:
    - The source text is an indifferent form of Arabic since the language of the Holy Quran is considered to be classical Arabic. This Quranic corpus differs from the modern standard Arabic used today, in terms of word-forms and language structures [31].

- The text of the Quranic corpus contains diacritics that are essential in the way of word vocalization.
- The traditional grammar of (الإعراب) is also used in this Quranic corpus.

4) *Quran index corpus*: This Quran index corpus is a dataset that is manually annotated and processed [29]. It is basically built to be used in Alfanous Quranic search engine [32].

As borne out by the statistics of used data sets reported in Tab. 1, almost all notes will be discussed as follows. NAFIS corpus contains few words [27]. It only has 173 Arabic words, which are not well-chosen in a way to cover different cases and aspects of our daily life. The Arabic Golden corpus is cited as 'Gold' corpus, and it has more words than the NAFIS corpus. However, the Gold corpus has a poor number of unrepeated roots and lemmas (59 lemmas and 36 roots). This will lead to an increase in average of frequency of the words by roots to be 32.36 over other datasets (in NAFIS 1.27, Quranic 8.21, and in Quran index corpus 6.72). The frequency of word by lemma in the Gold corpus is also in high average equal 19.75, while the average of lemmas in Quranic corpus is 3.51, in Quran index corpus is 3.08, and in NAFIS corpus is 1.28.

Table 1: Statistics and main features of used data sets

Main Features	Gold Arabic Corpus	NAFIS Corpus	Quranic Arabic Corpus	Quran Index Corpus
Words count	1,165	173	15,605	15,038
Unique words count	1,154	153	14,886	14,870
Unique lemmas count	59	129	4.450	4.884
Unique roots count	36	131	1,901	2,147
Avg.: mean of words by lemma	19.75	1.28	3.51	3.08
Avg.: mean of words by root	32.36	1.27	8.21	6.72
Max words by lemma	113	5	101	85
Max words by root	130	5	202	155

Our experimental investigation found that the Quranic corpus data is not suitable to do tests, and it needs more preprocessing. This problem in the data of Quranic corpus is due to that the texts used a classical Arabic language, unlike the Quran index. Further, the roots in this Quranic corpus are not normalized, and the affixes are listed in separate rows. Therefore, we perform a suitable preprocessing to revise this problem in the Quranic corpus to be ready for experimental tests.

In the summary, NAFIS corpus is a relatively small dataset. Gold corpus has more number of words, but with a higher frequency of words by lemma and root. The quranic corpus has classical Arabic language words. All the mentioned limitations will influence the final quality of the testing results. Consequently, full explanation and discussion is presented with deep analysis in the next subsection, after performing wide evaluation experiments and comparative tests.

5.2 Comparative Evaluation Tests

In this section, the effect of developing our proposed enhanced stemming algorithm is thoroughly explained with a wide range of experimental tested results. The proposed enhanced stemmer provides extraction of stem and root at the same time, while other Arabic light stemmers provide only one extraction at the same time. Khoja-stemmer and ISRI-stemmer primarily act as rooters, while FARASA operates as a stemming algorithm. However, Assem-stemmer was developed based on the SnowBall algorithm, which allows one extraction at a time. Thus, Assem stemmer is developed in two separate packages, to be configured as stemmer or as rooter. For this reason, the results of our enhanced stemmer is compare with obtained results from other stemmers in two different tasks, which are stem extraction and root extraction as follows.

5.2.1 Stem Extraction Tests

To evaluate the proposed enhanced algorithm as a stemmer, the same four datasets containing the roots, stems, and lemmas columns are used. The columns of stems and lemmas will be used to compare the results of stems that are obtained from our proposed enhanced stemmer against the obtained results from other existing stemmers (Khoja stemmer [6], ISRI stemmer [7], Assem SnowBall-based stemmer [9], FARASA stemmer [8], and Tashaphyne base stemmer [25]). All obtained results are calculated based on the measurements of accuracy metrics that are adapted from [33, 2]. The experimental results proved that ISRI and Khoja are mostly acting as rooters than as stemmers. Nevertheless, we use them here in the stem extraction tests, due to that both ISRI and Khoja present stem by extracting the minimal part of the input word. Therefore, many researchers in the literature cited ISRI and Khoja as stemmers.

Table 2: Accuracy rate for stem extraction results

	Gold Arabic Corpus	**NAFIS Corpus**	**Quranic Arabic Corpus**	**Quran Index Corpus**
Proposed enhanced stemming algorithm	63.95%	**74.71%**	48.01%	**59.34%**
Tashaphyne base algorithm (2018)	37.85%	41.38%	33.73%	39.99%
Assem stemming algorithm (2018)	**71.07%**	59.77%	48.98%	52.27%
FARASA stemming algorithm (2016)	44.38%	68.39%	**53.96%**	59.06%
ISRI stemming algorithm (2005)	16.82%	47.13%	32.73%	35.86%
Khoja stemming algorithm (1999)	13.48%	28.16%	30.17%	34.25%

The stem evaluation results in terms of linguistic accuracy are reported in Tab. 2. Our proposed enhanced light stemmer got the best stems accuracy results for NAFIS and Quran index datasets, with accuracy rates of 74.71% and 59.34%, respectively. For Gold corpus, Assem-stemmer obtained the best score with a 71.07% accuracy rate, while the proposed enhanced stemmer got the second score with a 63.95% accuracy rate. Further, FARASA-stemmer obtained the best accuracy rate 53.96%, respect to the Quranic corpus. Meanwhile, our enhanced stemmer also obtained the third score of 48.01% accuracy rate for this Quranic corpus, after Assem stemmer.

5.2.2 Root Extraction Tests

The proposed enhanced light stemming algorithm operates the stemming and rooting processes without using the root lookup dictionary, which acts like ISRI-stemmer. Meanwhile, ISRI-stemmer uses the Khoja algorithm but without using the root lookup dictionary. The obtained results are calculated based on the accuracy metrics adapted from [2, 34]. Basically, the effect of proposed enhanced algorithm is tested as a rooter against other existing Arabic light stemmers, which are Khoja stemmer [6], ISRI stemmer [7], Assem SnowBall-based stemmer [9], FARASA stemmer [8], and Tashaphyne stemmer [25].

Khoja stemmer extracts the root from the input word with dictionary verification, while ISRI stemmer applied the same Khoja algorithm but without using root dictionary verification step. Assem based on Snowball is presented in two variants. The first one is a rooter, while the second is a stemmer. Therefore, we cite and compare with Assem stemmer indifferently. Thus, Assem acts as rooter when it extracts roots, and Assem as stemmer when it extracts stems.

Table 3: Accuracy rate for root extraction results

	Gold Arabic Corpus	**NAFIS Corpus**	**Quranic Arabic Corpus**	**Quran Index Corpus**
Proposed enhanced stemming algorithm	**73.82%**	56.90%	**63.69%**	**60.75%**
Tashaphyne base algorithm (2018)	57.51%	**71.26%**	38.94%	42.96%
Assem stemming algorithm (2018)	70.82%	59.77%	55.28%	56.02%
FARASA stemming algorithm (2016)	7.73%	56.32%	23.05%	25.04%
ISRI stemming algorithm (2005)	57.68%	54.02%	50.44%	50.17%
Khoja stemming algorithm (1999)	65.24%	59.77%	58.40%	57.23%

Consequently, the proposed new enhanced stemming algorithm recorded the best precision values for extracting more accurate roots. As seen in Tab. 3, our enhanced stemmer got the highest linguistic accuracy on three used datasets (Gold 73.82%, Quranic corpus 63.69%, Quran index 60.75% respectively). For the fourth data (i.e., NAFIS corpus), the Tashaphyne basic stemmer obtained the highest linguistic accuracy, which is 71.26%. Noted that the Khoja-stemmer and Assem as rooter, obtained the same accuracy rate 59.77% only on NAFIS corpus. Then, our proposed enhanced stemmer reported a 56.90% accuracy rate, which is in stage four after those of Tashaphyne, Khoja, and Assem stemmers.

6 Conclusions and Future Work

In this research paper, a new enhanced Arabic light stemming algorithm is presented for stems and roots extraction in the information retrieval domain. The proposed stemming algorithm is enhanced with a new adaptation of direct acyclic graph (DAG) model. This adapted DAG model is collaborated with multiple features to extract strong initial segments from the input data. The proposed enhanced light stemming algorithm can be used as a rooter and as a stemmer, according to the affixes list that is predefined and customized. Therefore, the main algorithm of proposed stemmer extracted initial outputs, and then iteratively processed to enhance the choices of stems and roots outputs until the best results are finally obtained.

For comparative evaluation purposes, four different data sets are used, which are Gold Arabic Corpus, NAFIS, Quranic Arabic corpus, and Quran index corpus. Our proposed enhanced stemming algorithm with deploying the DAG model, builds the initial segments (i.e., stem-hits), and then achieves the best results of final stems and roots. Furthermore, the outputs of our proposed enhanced stemmer are compared with those obtained from five other well-known light stemmers (Khoja, ISRI, Assem, FARASA, and Tashaphyne stemmer), using the same four data sets. Finally, this comparative process proved that our proposed enhanced stemming algorithm can excel the other five existing stemmers for almost all used datasets.

Extracting strong initial segments from the input word, has a direct impact and leads to more accuracy on the final outputs. Therefore, further study to investigate and adapt other segmentation models more efficiently than the DAG model will improve the final results. Furthermore, the effect of extraction techniques (such as stem extraction and root extraction) in improving the outcomes of the stemming algorithm, should also be studied and deployed with more robustness techniques in the future to enhance the extraction process of stemming. Many challenging in existing used data sets can be also analyzed to further preparing new valid and more comprehensive corpus in future work.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] P. Willett, “The Porter stemming algorithm: then and now,” *Program*, vol. 40, no. 3, pp. 219–223, 2006.

[2] A. Wahbeh, M. Al-Kabi, Q. Al-Radaideh, E. Al-Shawakfa and I. Alsmadi, “The effect of stemming on Arabic text classification: an empirical study,” *International Journal of Information Retrieval Research*, vol. 1, no. 3, pp. 54–70, 2011.

[3] H. Rashaideh, A. Sawaie, M. A. Al-Betar, L. M. Abualigah, M. M. Al-laham et al., “A Grey Wolf Optimizer for Text Document Clustering,” *Journal of Intelligent Systems*, vol. 29, no. 1, pp. 814–830, 2020.

[4] M. M. Ali, M. M. A. Shquier, A. S. Eldeen, M. E. Zidan and R. M. Al-Khatib, “Novel approach in multilingual and mixed English-Arabic test collection,” *International Journal of Computing Science and Mathematics*, vol. 11, no. 3, pp. 291–304, 2020.

[5] J. Suzuki, T. Hirao, Y. Sasaki and E. Maeda, “Hierarchical directed acyclic graph kernel: Methods for structured natural language data,” in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, Sapporo, Japan, pp. 32–39, 2003.

[6] S. Khoja and R. Garside, “Stemming Arabic text,” *Lancaster*, UK, Computing Department, Lancaster University, 1999.

[7] K. Taghva, R. Elkhoury and J. Coombs, “Arabic stemming without a root dictionary,” in *Information Technology: Coding and Computing (ITCC 2005)*, Las Vegas, NV, USA, 152–157, 2005.

[8] A. Abdelali, K. Darwish, N. Durrani and H. Mubarak, “FARASA: A fast and furious segmenter for Arabic,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, San Diego, California, pp. 11–16, 2016.

[9] A. Chelli, “Assem's Arabic light stemmer,” 2018. [Online]. Available: https://arabicstemmer.com/.

[10] K. C. Ryding, “Arabic word structure: an overview,” *Cambridge: Cambridge University Press*, pp. 44-56, 2005, https://doi.org/10.1017/CBO9780511486975.004.

[11] K. M. O. Nahar, R. M. Al-Khatib, M. Al-Shannaq, M. Daradkeh and R. Malkawi, “Direct text classifier for thematic Arabic discourse documents,” *International Arab Journal of Information Technology*, vol. 17, no. 3, 2020.

[12] M. Aronoff, “Morphology by itself: stems and inflectional classes,” in *MIT press*, 1994. [Online]. Available: https://pascal-francis.inist.fr/vibad/index.php?action=getRecordDetail&idt=777867.

[13] K. C. Ryding, “Arabic: a linguistic introduction,” *Cambridge: Cambridge University Press*, 2014, https://doi.org/10.1017/CBO9781139151016.

[14] L. Al Qassem, D. Wang and H. Barada, “Noun extraction tool for ANLP applications,” in *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, Laguna Hills, CA, USA, pp. 308–309, 2018.

[15] M. ElDefrawy, Y. El-Sonbaty and N. A. Belal, “CBAS: context based Arabic stemmer,” *International Journal on Natural Language Computing*, vol. 4, no. 3, pp. 1–12, 2015.

[16] E. Loper and S. Bird, “NLTK: The natural language toolkit,” in *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics*, Philadelphia, Pennsylvania, USA, pp. 63–70, 2002.

[17] M. F. Porter and R. Boulton, “Snowball stemmer,” 2001. [Online]. Available: https://snowballstem.org/.

[18] M. N. Al-Kabi, S. A. Kazakzeh, B. M. A. Ata, S. A. Al-Rababah and I. M. Alsmadi, “A novel root based Arabic stemmer,” *Journal of King Saud University - Computer and Information Sciences*, vol. 27, no. 2, pp. 94–103, 2015.

[19] M. Al-Ayyoub, A. A. Khamaiseh, Y. Jararweh and M. N. Al-Kabi, “A comprehensive survey of Arabic sentiment analysis,” *Information Processing & Management*, vol. 56, no. 2, pp. 320–342, 2019.

[20] A. Blumer, J. Blumer, D. Haussler, A. Ehrenfeucht, M. T. Chen et al., “The smallest automation recognizing the subwords of a text,” *Theoretical Computer Science*, vol. 40, pp. 31–55, 1985.

[21] R. Alur, S. Chaudhuri and P. Madhusudan, "Languages of nested trees," in *International Conference on Computer Aided Verification (CAV 2006)*, Seattle, WA, USA, pp. 329–342, 2006.

[22] M. Aljlayl and O. Frieder, "On Arabic search: improving the retrieval effectiveness via a light stemming approach," in *Proceedings of the eleventh international conference on Information and knowledge management (CIKM'02), Mclean, Virginia, USA*, pp. 340–347, 2002.

[23] Y. Jaafar and K. Bouzoubaa, "A survey and comparative study of Arabic NLP architectures," in *Intelligent Natural Language Processing: Trends and Applications*, Springer, pp. 585– 610, 2018.

[24] W. Cherif, A. Madani and M. Kissi, "A new modeling approach for Arabic opinion mining recognition," in *Proceedings of 2015 Intelligent Systems and Computer Vision (ISCV)*, Fez, Morocco, pp. 1–6, 2015.

[25] T. Zerrouki, "Tashaphyne 0.3.2, Arabic light stemmer," 2018. [Online]. Available: https://pypi.org/project/Tashaphyne/.

[26] B. Lakhdar, "Golden Arabic corpus," 2018. [Online]. Available: https://github.com/ibnmalik/golden-corpus-arabic.

[27] D. Namly, R. Tajmout, K. Bouzoubaa and L. Abouenour, "NAFIS: a gold standard corpus for Arabic stemmers evaluation," in *Proceedings of the 28th International Business Information Management Association Conference (28th IBIMA)*, Seville, Spain, 2016.

[28] K. Dukes, "The Quranic Arabic corpus," 2009. [Online]. Available: https://corpus.quran.com/.

[29] A. Chelli, A. Balla, and T. Zerrouki, "Advanced search feature in noble Quran," in *Proceedings of 2013 Taibah University International Conference on Advances in Information Technology for the Holy Quran and Its Sciences*, Madinah, Saudi Arabia, pp. 681–699, 2013.

[30] K. M.O. Nahar, R. M. Al-Khatib, M. A. Al-Shannaq and M. M. Barhoush, "An efficient holy Quran recitation recognizer based on SVM learning model," *Jordanian Journal of Computers and Information Technology*, vol. 6, no. 4, pp. 395–417, 2020.

[31] K. M.O. Nahar, M. A. Al-Shannaq, R. Alshorman, R. M. Al-Khatib and M. A. Ottom, "Handicapped wheelchair movements using discrete Arabic command recognition," *Scientific Journal of King Faisal University Basic and Applied Sciences*, vol. 21, no. 1, pp. 171–184, 2020.

[32] A. Chelli, "Alfanous: advanced Quranic search engine," 2018. [Online]. Available: https://launchpad.net/alfanous.

[33] Y. Jaafar, K. Bouzoubaa, A. Yousfi, R. Tajmout and H. Khamar, "Improving Arabic morphological analyzers benchmark," *International Journal of Speech Technology*, vol. 19, no. 2, pp. 259–267, 2016.

[34] Y. Jaafar, D. Namly, K. Bouzoubaa and A. Yousfi, "Enhancing Arabic stemming process using resources and benchmarking tools," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 2, pp. 164–170, 2017.