Intelligent Information Management, 2017, 9, 39-67
http://www.scirp.org/journal/iim
ISSN Online: 2160-5920
ISSN Print: 2160-5912

A Comparative Survey on Arabic Stemming: Approaches and Challenges

Mohammad Mustafa[1], Afag Salah Eldeen[2], Sulieman Bani-Ahmad[3], Abdelrahman Osman Elfaki[4]

[1]Department of Computer Information Systems, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, SA
[2]Department of Computer Science, College of Computer Science and Information Technology, Sudan University of Science and Technology, Khartoum State, Sudan
[3]Department of Computer Information Systems, School of Information Technology, Al-Balqa Applied University, Salt, Jordan
[4]Department of Information Technology, Faculty of Computers and Information Technology, University of Tabuk, Tabuk, Saudi Arabia
Email: mmustafa@ut.edu.sa, afagsalah@hotmail.com, sulieman@bau.edu.jo, a.elfaki@ut.edu.sa

How to cite this paper: Mustafa, M., Eldeen, A.S., Bani-Ahmad, S. and Elfaki, A.O. (2017) A Comparative Survey on Arabic Stemming: Approaches and Challenges. *Intelligent Information Management*, **9**, 39-67.
https://doi.org/10.4236/iim.2017.92003

Received: February 21, 2017
Accepted: March 28, 2017
Published: March 31, 2017

Abstract

Arabic, as one of the Semitic languages, has a very rich and complex morphology, which is radically different from the European and the East Asian languages. The derivational system of Arabic, is therefore, based on roots, which are often inflected to compose words, using a spectacular and a relatively large set of Arabic morphemes affixes, e.g., antefixs, prefixes, suffixes, etc. Stemming is the process of rendering all the inflected forms of word into a common canonical form. Stemming is one of the early and major phases in natural processing, machine translation and information retrieval tasks. A number of Arabic language stemmers were proposed. Examples include light stemming, morphological analysis, statistical-based stemming, N-grams and parallel corpora (collections). Motivated by the reported results in the literature, this paper attempts to exhaustively review current achievements for stemming Arabic texts. A variety of algorithms are discussed. The main contribution of the paper is to provide better understanding among existing approaches with the hope of building an error-free and effective Arabic stemmer in the near future.

Keywords

Arabic Language, Light Stemming, Root-Based Stemming, Co-Occurrence, Artificial Intelligence Stemming

1. Introduction

The major task of an Information Retrieval (IR) system is how to match between a searchable document representation (documents) and a user need, which is always expressed in terms of queries. The process of representing documents, in which keywords or terms are extracted, is called indexing. Indexing often goes through several operations, most of which are language-dependent. Among these operations, stemming stands as one of the major steps that every IR system must handle. Since documents and/or queries may have several forms of a particular word, stemming is the process of mapping and transforming all the inflected forms of that word into a common, shared and canonical form and, thereby, this canonical form would be the most appropriate form for indexing and for searching, as well. In other words, stemming renders different inflected and variant forms of a certain word to a single word stem. In monolingual IR, stemming appears to have a positive impact on recall more than precision [1]. This means that stemming helps to find more relevant documents but it is not able to provide the best ranking for the retrieved list.

Over the last decades, Arabic has become one of the popular areas of research in IR, especially with the explosive growth of the language on the Web, which shows the need to develop good techniques for the increasing contents of the language. This increasing interest in Arabic, however, is caused by its complex morphology, which is radically different from the European and the East Asian languages [2]. In addition, Arabic has complicated grammatical rules and it is very rich in its derivational system [3]. These features make the language challenging in computational processing and morphological analysis because in most cases, exact keyword matching between documents and user queries, is inadequate.

A number of studies have been devoted to stemming for a wide range of languages, including Arabic. Different approaches were proposed. For Arabic stemming [3] [4], examples include light stemming, morphological analysis, statistical-based stemming using co-occurrence analysis, N-grams or parallel corpora (collections). Some of these stemming approaches, especially those statistical ones, are language-dependent and are not tailored to Arabic only, while others provide more language independency. It is reported that stemming has a high positive effect on highly inflected languages, such as Arabic [5].

Among these techniques, two major approaches are the most dominant for Arabic stemming. These are light stemming (known also as affix removal stemming) and heavy stemming (morphological analysis stemming). The light stemming chops off some affixes—such as plural endings in English lightly from words, whereas the second technique, which is heavy stemming, performs heuristic and linguistic processes so as to extract the root of the word, the possible roots or the stem of the word. The stem in Arabic IR is the least form of the word without any prefixes and suffixes, whereas the root of the surface form is the basic unit which often consists of three letters. Technically, root base stemmers attempt always to analyze words and to produce their roots.

Other techniques such as the use of corpus-based statistics and lexicons (to determine most frequent affixes and employing genetic algorithms and neural networks) have been also reported in the literature. Approaches like co-occurrence techniques for clustering words together and the use of parallel corpora have been also investigated.

However, in spite of the significant achievements and developments of these Arabic stemming techniques, each of the proposed approaches has some pros and cons and it is yet unclear which technique is to be adopted for indexing and/or stemming Arabic texts.

This paper attempts to review current techniques to Arabic stemming problem. It provides firstly a comprehensive examination to the features of the Arabic that make the language challenging to Natural Language processing (NLP) and Information Retrieval (IR). The paper also compares among a considerable number of stemmers and how each of them works and produces the stem and/or root from Arabic text. The strengths and the weaknesses of each technique are also provided.

The rest of this paper is organized as follows. Section two introduces the characteristics of Arabic language which makes it challenging to Arabic IR task. Section three is an in-depth coverage for the existing approaches to Arabic stemming. Several studies are presented in this section. In section four an intensive discussion on the current approaches and their limitations is conducted. In section five, the paper is concluded.

2. Why Arabic Is Challenging

Arabic is one of the Semitic languages, which also includes Hebrew, Aramaic and Amharic. It is the lingua-franca of a large group of people. It is estimated that there are approximately four hundred million first-language speakers of Arabic [3] [6]. Since it is the language of religious instruction in Islam, many other speakers from varied nations have at least a passive knowledge of the language. Arabic also is one of the six official languages of the (UN) and it is the fifth most widely used language in the world [2] [7].

Sentences in Arabic are delimited by periods, dashes and commas, while words are separated by white spaces and other punctuation marks. Arabic script is written from right-to-left while Arabic numbers are written and read from left-to-right. Script of Arabic consists of two types of symbols [3] [8]: these are the letters and the diacritics (known also as short vowels), which are certain orthographic symbols that are usually added to disambiguate Arabic words. Cited in [2], Tayli and Al-Salamah stated that the Arabic alphabet has 28 letters, and, unlike English, there is no lower and upper case for letters in Arabic. An additional character, which is the HAMZA (ء), has been also added, but, usually it is not classified as the 29th letter.

Arabic words are classified into three main parts-of-speech: nouns (including adjectives and adverbs), verbs and particles. Particles in Arabic are attached to verbs and nouns. Words in Arabic are either masculine or feminine. The feminine is often formed differently from the masculine, e.g., مُبرمج and مُبرمجة (meaning: single masculine programmer and single feminine programmer, respectively). The same feature appears also in both nouns and verbs in literary Arabic in order to indicate number (singular, dual "for describing two entities" and plural) as in مُبرمج ,مُبرمجان and مُبرمجون (meaning: singular programmer, two programmers and more than two programmers, respectively).

Arabic has a complex morphology. Its derivational system is based on 10,000 independent roots [9]. Roots in Arabic are usually constructed from 3 consonants (tri-literals) and it is possible that 4 consonants (quad-literals) or 5 consonants (pent-literals) are used. Out of the 10,000 roots, only about 1200 are still in use in the modern Arabic vocabulary [10]. Words are formed by expanding the root with affixes using well-known morphological patterns (known sometimes as measures). For example, **Table 1** shows some different forms derived for the word أخلاء, which is the plural of the word خليل (meaning: a close friend) after being attached to different affixes. All words are correct in Modern Standard Arabic (MSA). This feature causes Arabic to have more words that can occur only once in text, compared to other languages, e.g., English [2] [11].

Words and morphological variations are derived from roots using patterns. Grammatically, the main pattern, which corresponds to the tri-literal root, is the pattern فَعَل (transliterated as f-à-l). More regular patterns, adhering to well-known morphological rules, can be derived from the main pattern فعل (f-à-l). Examples of some patterns are فَعَل، فِعَال and أَفَاعِيل, transliterated as f-à-l, f-i-à-l and a-f-à-i-l, respectively.

Different kinds of affixes can be added to the derived patterned words to construct a more complex structure. Definite articles—like ال (its counterpart is the definite "the"), conjunctions, particles and other prefixes can be affixed to the beginning of a word, whereas suffixes can be added to the end. For example, the word لنجْمَعنّهم (meaning: we will surely gather them) can be decomposed as follows: (antefix: ل, prefix: ن, root: جمع, suffix: ن and postfix: هم). For the purpose of understanding stemming, all Arabic affixes are listed in **Table 2**, quoted in Kadri and Nie [12].

Antefixes, whether they are separated or not, are usually prepositions added to the beginning of words before prefixes. Prefixes are attached to exemplify the present tense and imperative forms of verbs and usually consist of one, two or three letters. Suffixes are added to denote gender and number, for examples in dual feminine and plural masculine. Postfixes are used to indicate pronouns and to represent the absent person (third person), for example. Usually this morphology is used to create verbal and nominal phrases. **Table 3** illustrates several lexical words derived from the root حسب, which corresponds to the main pattern

Table 1. Different affixes attached to Arabic word **أخلاء** (meaning: the plural of the word **خليل**, which means "a close friend").

Word
أخلاء
أخلائه، أخلاؤه، أخلاءه، أخلائهم، أخلاءهم، أخلاؤهم، أخلائهم، أخلائهن، أخلائهما، أخلاؤهما، أخلاءنا، أخلائنا، أخلاؤنا، أخلائكم، أخلائك، أخلاءك، أخلاؤها، أخلاؤها، أخلائها،أخلائي، وأخلائي، الأخلاء، بالأخلاء، بأخلاء، بأخلائهم ... إلخ

Table 2. Affixes in MSA (Arabic is read from right to left).

Antefixes	Prefixes	Suffixes	Postfixes
وبال، وال، بال، فال، كال، ولل، ال، وب، ول، لل، فس، فب، فل، وس، ك، ف، ب، ل	ا، ن، ي، ت	تا، وا، ين، ون، ان، ات، تان، تين، يون، تما، تم، و، ي، ا، ن، ت، نا، تن	ي، ه، ك، كم، هم، نا، ها، تي، هن، كن، هما، كما
Prepositions meaning respectively: and with the, and the, with the, then the, as the, and to (for) the, the, and with, and to (for), then will, then with, then to (for), and will, as, then, and, with, to (for)	Letters meaning the conjugation person of verbs in the present tense	Terminations of conjugation for verbs and dual/plural/female/male marks for nouns	Pronouns meaning respectively: my, his, your, your, their, our, her, my, their, your, their, your

Table 3. Different derivatives from the root حسب..

Arabic Word	Pattern Transliterated	Meaning
حسب	*f-à- l*	Compute (a tri-literal root)
يحسب	*y- f-à- l*	He computes
حسبنا	*f-à- l-n-a*	We compute
حسبن	*f-à- l-n*	They compute (plural feminine)
يحسبون	*y- f-à- l-o-n*	They compute (plural masculine)
حسبا	*f-à- l-a*	They compute (dual masculine)
حاسوب	*f-a-à-o- l*	Computer (Machine name)
حسّب	*f-à- à- l*	He computes (for intensifying verbs)

فعل (f-à-l), according to some different patterns, in which some letters are added to the main pattern.

Affixes in Arabic may include also some clitics. Clitics, which have been used in the proposed stemmers and can be proclitics or enclitics according to their locations in words, are morphemes that have the syntactic characteristics of a word but are morphologically bound to other words [13]. Thus, clitics are attached to the beginning or end of words. Such clitics include some prepositions, definite articles, conjunctions, possessive pronouns, particles and pronouns. Examples of clitics are the letters ك (pronounced as KAF) and ف (pronounced as FAA), which mean as and then, respectively.

Arabic also has three grammatical cases, as well. These cases are: nominative, accusative and genitive. For example, if the noun is a subject, then it will have the nominative grammatical case; if it is an object, the noun will be in the accusative case; and the noun will be in a genitive case if it is an object for a preposition. These grammatical cases cause Arabic to derive many words from a single noun (*i.e.* adjective) because it often results in a different form of the word. Note that adjectives in Arabic are nouns. For example, the different forms that can be derived from the adjective مزارع (meaning: farmer) according to their both grammatical forms may include words like: مزارعة (for singular feminine in nominative, accusative and genitive cases), مزارعان (for dual masculine in nominative case), مزارعَين (for dual masculine in accusative and genitive cases), مزارعتان (for dual feminine in nominative case), مزارعتين (for dual feminine in accusative and genitive cases), مزارعون (plural masculine in nominative case), مزارعِين (for plural masculine in accusative and genitive cases) and مزارعات (for plural feminine in nominative, accusative and genitive cases).

Morphology adds a level of ambiguity that makes the exact keyword matching mechanism inadequate for retrieval. Morphological ambiguity can appear in several cases. For example, clitics may accidentally produce a form that is homographic or homogenous (the same word with two or more different meanings) with another full word [2] [3] [14]. For example, the word علم (meaning: science) can be joined with the clitic (ي) to construct the word علمي (meaning: my knowledge) which is homographic with the word علمي (meaning: scientific). Additionally, Arabic grammar contributes to the morphological ambiguity. For example, according to some Arabic grammar rules, sometimes vowels are removed from roots. The set of the vowel letters in Arabic consists of three letters: ALIF, YAA and WAW (أ، ي، و). These letters have different rules that do not obey the derivational system of Arabic and make them very changeable. For instance, the last letter YAA is removed in a word like امشي (meaning: go), resulting in امش, if it appears in an imperative form.

Besides the complex morphology, Arabic also has a very complex type of plurals known as broken plural. Plurals in Arabic do not obey morphological rules. They are similar to cases like: *corpus* and *corpora*; and *mouse* and *mice* in English, but differing in that there is no rule-based morphological syntax to the broken plurals. Broken plurals constitute 10% of Arabic texts and 41% of plurals [2] [15]. Unlike English, the plural in Arabic indicates any number higher than two. The term broken means that the plural form does not resemble the original singular form. For example, the plural of the word نهر (meaning: river) is أنهار (rivers). In the simple cases of broken plurals, the new inflected plural has some letters in common when it is compared to the singular form, as in the previous example. But in many cases the plural is totally different from the original word, e.g., the plural of the word إمراة (meaning: woman) is نساء (women).

Diversity in broken plurals makes them highly unpredictable. In most cases knowing the singular form does not assist to deduce the plural, and vice-versa. This fact shows how much broken plurals lead to a mismatch problem in Arabic IR.

Arabic also has very diverse types of orthographic variations. They are very common and present real challenges for both Arabic IR and NLP systems. Examples include, but they are not limited to *Typographical Variations*, which merely caused by the Arabic letters ALIF with its different glyphs (أ, إ, آ and ا) and YAA with its dotted and un-dotted forms (ي and ى) and HAA with the forms ه and ة. In most cases, one of the glyphs of a certain letter is altered/dropped, initially, medially or finally, with another glyph of the same letter when writing text [16]. **Table 4** shows some examples of different typographical variations in MSA. Sometimes the typographical variant changes the meaning of the original word significantly, for example the قرآن (meaning: the Holy Quran) is typographically changed to قران (meaning: marriage contract), when the letter ALIF MADDA glyph in the middle is changed to bare ALIF.

3. Stemming in Arabic

Since Arabic is an inflectional language, a large number of studies have been devoted to the analysis of the best approach to index Arabic words. The process of producing index terms often goes through several operations, most of which are language-dependent. *Normalization* and *stemming* are among these major processes.

Normalization is the process of producing the canonical form of a token and/or a word in order to maximize matching between a query token and document collection tokens. In its simple form normalization pre-processes tokens to a single form, but very lightly. This is often done in several pre-processing stages so as to render different forms of a particular letter to a single Unicode representation, e.g., replacing the Arabic letter un-dotted ى with a final dotted ي, when this letter appears at the end of an Arabic word.

In its complex forms, normalization is used to handle morphological variation and inflation of words [17]. This is called stemming. Stemming is the process of rendering different inflected and variant forms of a certain word to a single term, known as *stem*. For instance, words like participating, participates, participation and participant may all be rendered to a common single stem *participat*.

Since documents and/or queries may have several forms of a particular word, stemming should map and transform all the inflected forms of a word into a common shared form and, thereby, this shared form would be the most appropriate form for indexing the representations of documents and for searching as well.In monolingual IR, stemming appears to have a positive impact on recall more than precision [5]. Furthermore, stemming shows a high positive effect on highly inflected languages, such as Arabic [5]. An additional advantage for the

Table 4. Illustrates some examples for typological variants in Arabic.

MSA	Variant	Gloss	Typographical Occurrence
امتحان	إمتحان	Exam	The final bare ALIF is changed to ALIF HAMZA below
صفاء	صفا	Purity	The final HAMZA is dropped
قرآن	قران	The Quran	ALIF MADDA in the middle is altered to bare ALIF
علاء	علا	A proper noun	They compute (plural feminine)
نافذه	نافذة	Window	The final letter HAA is altered to a different letter, which is TAA MARBOOTA
زراعي	زراعى	Agricultural	The final dotted YAA is changed to un-dotted YAA

stemming is that it also reduces the size of the index since many words are grouped together in a single canonical form.

In Arabic IR, the word is the surface form which is often obtained by tokenizing the text (*i.e.* tokenizing text on white space and punctuations). Thus, the word in Arabic in its complete structure is a concatenated form of letters consisting of prefixes, morpheme and suffixes, e.g., وألعابهم (meaning: and their games or their toys). From that perspective, the issue of whether Arabic index terms should be roots or stems has always been a major question. Cited in [13], some studies claimed that the lemmatized form of words in Arabic is the stem, while others argue that the lemma of the language is the root and the stem is only a manifestation to the root. By the term lemma [1] [3], it is meant the single dictionary entry form of the several inflected derivatives of a word. Nevertheless, there is an implicit assumption in NLP and IR that the stem in Arabic IR is the least form of the word without any prefixes and suffixes or their attached clitics, but possibly having extra letters medially. In the case of verbs in Arabic language, this is often the third person, perfective (past) and singular forms of verbs, whereas the stem is the singular form in the case of nouns (including adjectives). For instance, the stem of the word وألعابهم above is ألعاب in which both prefixes and suffixes from the beginning and ending of the word is truncated.

On the other hand, it is known in Arabic linguistics community that the root of the Arabic surface form is the basic unit, which usually rhymed and/or patterned by the pattern فعَل as it was described earlier. Accordingly, if an Arabic root is to be extracted from a surface form, all the affixes that appear in that word, even they are written medially, should be stripped-off.

Accordingly indexing Arabic words has two different paradigms [3] [13] [14]: either to index stem or root. Stem indexing paradigm attempts to remove only a few common numbers of prefixes and suffixes from words and without attempting to identify the patterns of words or their roots. On the other hand, root indexing technique attempts to analyze the words, which often contain root, patterns, prefixes and suffixes, so as to produce the root or all the possible roots of a word.

In order to achieve the goal of indexing the most adequate Arabic term (stem or root) from a word/token, several approaches have investigated from the use of lexicons and dictionaries to morphological analysis and combination of different techniques. Each method has its pros and cons and the studies investigated exhaustively what is the best technique to index Arabic words.

Due to large number of the studies in this specific area, researchers attempt to classify the techniques according to their algorithmic behaviors. Larkey, *et al.*, [4] clusters the techniques into four categories:

- Manually constructed dictionaries, in which words with their roots and their possible segmentations are stored in a large lookup table.
- Affix truncation techniques which often attempt to stem the words lightly by removing common suffixes and prefixes.
- Morphological analyzers, in which the root is extracted using morphological analysis.

- Statistical stemming which is based on clustering similar words in documents together.

In spite of the good classification of these techniques, but in the opinion of the authors this classification needs to be extended so as to include newer techniques. The new extended classification is shown in Figure 1.

Before delving into the details of each of the employed technique, it is important first to cover simple normalization. This is because stemming is in fact a complex normalization technique as it was illustrated earlier. In addition, the majority of the techniques perform some normalization technique firstly. Next sections explain normalization and stemming techniques in details.

3.1. Normalization

Before normalization, the majority of the Arabic stemming techniques process texts. Preprocessing in Arabic includes removal of non-characters, normalization of letters and removal of stopwords. Removal of non-characters [2] [18] includes the removal of punctuation marks, diacritics and *Kasheeda*, known also as *Tatweel*, which is an Arabic stylistic elongation of some words for cosmetic writing. For example, the word عادل (a proper noun) can be written with kasheeda as عـــــادل.

As it was shown earlier, normalization in Arabic is used to render different forms of a letter with a single Unicode representation. This is important to moderate the orthographic variations. Since there are only few Arabic letters that are the sources for orthographic variations of words, most stemming approaches handle them in a similar way. Accordingly, the majority of the stemming techniques normalize documents and queries using some or all of the following normalization [2] [12] [19].

Main Stemming Techniques for Arabic-Language Text								
Root-based and Morphological analysis	Affix truncation and light stemming		Statistical Stemming			Simple POS Tagging and Combined Techniques	Artificial Intelligence	
	Light Stemming	Linguistic Removal using corpus statistics	N-grams methods	Co-occurrence analysis	Clustering words using corpora		Neural networks	Genetic Algorithms

Figure 1. Classification of stemming techniques according to their algorithmic behaviors.

- Replacing ALIF in HAMZA forms (ALIF combined with HAMZA that is written above or below the ALIF like in “أ and إ”) and ALIF MADDA (آ) with bare ALIF (ا).
- Replacing final un-dotted YAA (ى) with dotted YAA (ي).
- Replacing final TAA MARBOOTA (ة) with HAA (ه).
- Replacing the sequence ىء with ئ.
- Replacing the sequence يء with ئ.
- Replacing ؤ with bare ALIF (ا).

In spite of the wide use of these normalization steps, Abdelali, *et al*., [18] stated that some of these normalizations may conceal word characteristics and create ambiguity. For instance, it is not always correct to unify all glyphs of ALIF to a plain ALIF as it may lead to invalid words. Similar trends were also shown by Daoud and Hasan [20] who showed that normalization of Arabic letters, especially in the middle of words can result in incorrect words. For instance, normalizing ALIF MADDA (آ) with bare ALIF (ا) in the Arabic word قرآن (meaning: the Quran) results in the word قران (meaning: marriage contract).

To address the impact of Arabic challenges on both monolingual and cross-lingual retrieval and the problem of orthographic resolution errors, such as changing the letter YAA (ي) to the letter ALIF MAKSURA (ى) at the end of a word, the studies in Xu, *et al*. [21] [22] used two different techniques to normalize spelling variations. The first technique is the normalization, which replaces all occurrences of the diacritical ALIF, HAMZA (إ،أ) and MADDA (آ), with a bare ALIF (ا). The second technique is the mapping, which maps every word with a bare ALIF to a set of words that can potentially be written as that word by changing diacritical ALIFs to the plain ALIF. All the mapped words in the set are equally probable, each of which obtains 1/n probability. The study of Xu and his team concluded that there is little difference between mapping techniques and normalization techniques for orthographic resolution.

The use of normalization techniques is almost similar in Arabic and it seems that in order to increase matching, the penalty paid is to normalize Arabic letters before stemming the words in which they occur.

3.2. Arabic Stemming Approaches

As it was illustrated earlier, we extended the classification of the employed approaches for stemming Arabic texts. The next section describes the techniques in this classification in details.

3.2.1. Root-Based and Morphological Analyzers

With the premise that the basic unit in Arabic is the root, root—based stemming technique attempts to perform heuristic and linguistic morphological analysis so as to extract the root of a word. For example, root-based algorithms produce the root عمل for the word وأعمالهم (meaning: and their works) because all affixes are removed. To achieve this goal of obtaining roots, researchers employ the use of Arabic morphological analyzers.

Khoja stemmer [23] is one of the most famous root-based stemmers. The algorithm was widely used in Arabic IR. It renders inflectional forms of words to produce their roots by removing their longest prefixes and suffixes, at first. For instance, the prefix ي and the suffix ون are firstly removed, using Khoja stemmer, if the input word is يلاعبون (meaning: they are playing with). The resulted word (in this case the word لاعب) is then matched with some predefined patterns and some list-driven roots. The selected pattern depends on the length of the extracted word. For example, for the word لاعب in our example the pattern فاعل may be chosen. By this matching process the root is produced as لعب (meaning: play) since the pattern فاعل is already predefined in the language that is has a bare letter ALIF (ا) added medially to the tri-literal pattern فعل. Finally, in the algorithm, the extracted root is compared to a list of roots to check its validity.

One advantage of Khoja stemmer is that it has the ability to detect letters that were deleted during the derivational process of words. For instance, the last letter YAA is removed in a word like امشي (meaning: go), resulting in امش, if it appears in an imperative form. As another example, the last letter ALIF in the root نما (meaning: grew) will be modified to WAW in the present form of this root and thus it will be نّمو instead of نّما. Using Khoja stemmer, it is possible to handle such cases.

However, in spite of its superiority and its wide use, the algorithm has a major drawback, that is the over-stemming in which the stemmer may erroneously cluster some semantically different words into a single root. This is because a tremendous number of Arabic words may have different semantic meanings although they share the same root, leading to low precision and high level of ambiguity. For example, both the words مقاتلات (meaning: fighters) and يتقاتلون (meaning: they are fighting each others) are originated from the canonical root قتل (meaning: to kill). Examples also include words like طفيليات (meaning: parasites) and لعوب (meaning: irresponsible) in which the produced roots using Khoja stemmer are طفل and لعب. Both stems are semantically different from the original word.

Additionally, sometimes the algorithm removes some affixes that are parts of words (known as mis-stemming), such as in the word مدرسه (meaning: schools) which will be stemmed to the root درس (meaning: lesson or learn in past tense). Khoja stemmer may also result in truncating some letters that are parts of the word. It is clear that removal of prefixes and suffixes blindly causes the stemmer to erroneously remove some original letters from the root. For instance, chopping-off suffixes and prefixes blindly from a word like بالغات (meaning: feminine adults) will result in removing the letters بال, which will be handled in the algorithm as a prefix although they are original letters of the root بلغ (meaning: to attain or to accomplish).

In his study for the Holy Quran, Hammo [24] stated that most of the failing cases of Khoja when it was used to stem words of the Holy book, were occurred when stemming proper names such as the names of Prophets, angels, ancient cities, places and people, numerals, as well as words with the diacritical mark shadda.

Darwish [25] developed Sebawai, a root-based analyzer that is based on automatically derived rules and statistics. Sebawai has two main modules. At first, a list of "word-root" pairs *i.e.* (وذهابهم, ذهب), which means (go, and they gone), had been constructed. The word-root pairs list was constructed using an old morphological analyzer called ALPNET. Then by comparing the root to the word, Sebawai extracts a list of prefixes, suffixes and stem templates. For example, given the pair (وذهابهم, ذهب) in the example above, the system produces و (meaning: and) as the prefix, هم (meaning: theirs) as the suffix and "CCAC" as the stem template (C's represent the letters in the root). During this phase of training, the frequency of each of the generated item (*i.e.* suffix) is computed and hence the probability that a prefix, suffix or stem template would occur is computed. For example, if the total number of occurrences of a certain prefix is 100, and the list of the generated word-root pairs is 1000, then a probability of value 0.1 is assigned to that prefix. As a result to this training phase probability tables are obtained for the suffixes, prefixes and stems of the training corpus (word-root pairs).

For the root detection phase, Sebawai takes the input word and produces all the possible combinations among prefix, suffix and template, which could result in forming that word. Once a possible combination is obtained, its product probability (with the independence assumption) is computed according to the previously estimated probabilities. The higher probability computed of a certain combination, its root is detected and matched against 10,000 roots to check its validity.

Sebawai has some limitations stated by its developer. First, it cannot stem transliterated words such as entity names (*i.e.*, انجلترا, which means England) because it binds the choice of roots to a fixed set. Second, Sebawai cannot deal with some individual words that constitute complete sentences, like لَنَهْدِيَنَّهُم (meaning: we will surely guide them) because the appearance of such words is very rare and thus, low probabilities are assigned. Additionally, since Sebawai is a root-based stemmer, it results in the same problem of over-stemming as in Khoja.

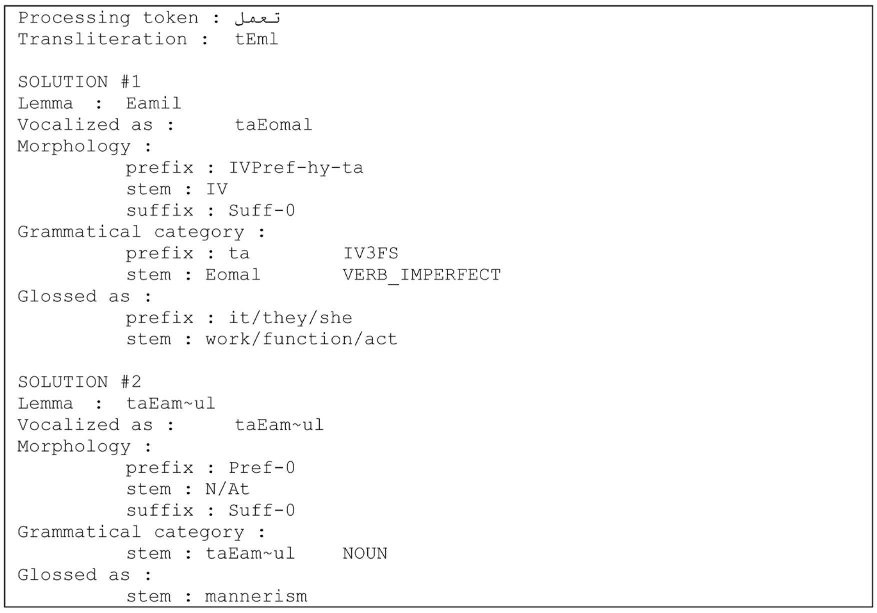
Buckwalter [26] developed a stem-based morphological analyzer which is one of the most popular and respected analyzers that were used widely in the TREC experiments.

Unlike, Khoja for example, Buckwalter produces a single stem or all the possible stems of the input word. The basic idea is similar to the one presented by Sebawi. At first, manually constructed tables are collected. The tables are based on three groups (prefixes, possible stems and suffixes). In addition, the valid combinations of each pair of the three groups (prefix/stem pairs, prefix/suffix pairs and stem/suffix pairs), are also stored in form of truth tables. Thus during the root detection phase, Buckwalter algorithm, which is coded in a program called Ara Morph, divided input word into three sub-strings (potential prefix, stems and suffix), with all its possibilities. The produced sub-strings are generated according to the pre-constructed tables. Following this, a matching process is performed for each possible combination of prefix, stem and suffix that could yield the input word. Hence using the truth tables pairs and if the first sub-string is a correct prefix, the second sub-string is a legitimate stem, the third sub-string is a legitimate sub-string and if the combination of all of them is valid then the second sub-string will extracted as a stem for the input word. If more than one stem is obtained then all of them will be listed.

Buckwalter is not just a stemmer. Instaed, it also tags the words with its possible POS and provides all the possible translations in English for that word. For example, for the word تعمل (tEml in Buckwalter transliteration), a version of the Buckwalter analyzer provided many solutions, two of them are presented in Figure 2.

One deficiency of Buckwalter's analyzer is that some words may not be stemmed because they may not be included in the stem table. In addition, broken plurals are not managed by the Buckwalter stemmer [21]. Attia [13] lists 11 cases where the Buckwalter analyzer failed to get their stems. One of the listed shortcomings is that Buckwalter failed to stem clitic question morpheme because of lack of coverage for such cases, e.g., أعادل (meaning: Is it correct that Adil).

Based on Buckwalter analyzer and the fact that the analyzer lists all the possible stems, Xu, *et al.*, [21] attempt to resolve ambiguity when more than one stem are returned. This is done by using a probabilistic model (as part of the retrieval task in that study) to accommodate ambiguity, which arises when equally probable probabilities are assigned to each of the obtained stems (when more than one stem is returned by the algorithm). Results showed that using one stem is somewhat better than using all the stems even they are in the IR task, but the improvement is not statistically significant. Abdelali [18] concluded that their approach may fail to eliminate ambiguous words. Since the same probability is assigned to both valid stem and possible stems, noise may be introduced.


```
Processing token : تعمل
Transliteration :  tEml

SOLUTION #1
Lemma  :  Eamil
Vocalized as :     taEomal
Morphology :
          prefix : IVPref-hy-ta
          stem : IV
          suffix : Suff-0
Grammatical category :
          prefix : ta         IV3FS
          stem : Eomal        VERB_IMPERFECT
Glossed as :
          prefix : it/they/she
          stem : work/function/act

SOLUTION #2
Lemma  :  taEam~ul
Vocalized as :     taEam~ul
Morphology :
          prefix : Pref-0
          stem : N/At
          suffix : Suff-0
Grammatical category :
          stem : taEam~ul     NOUN
Glossed as :
          stem : mannerism
```

Figure 2. Two solutions for the word تعمل using the Buckwalter.

Ghwanmeh, *et al*., [27] follows similar technique to Khoja to detect root. However, the algorithm is only used for those words whose lengths are greater than three letters. Accordingly, the algorithm takes the input word and leaves it as it appears if its length is less than four letters. Otherwise, the algorithm begins to remove the longest prefixes and suffixes and follows the she step by comparing the extracted stem to a list of pre-defined patterns. If the pattern length is equivalent to the generated stem, the algorithm chooses that pattern and extracts the root. The Algorithm was tested using a small dataset extracted from a small abstracts taken from Arabic proceedings of the Saudi conferences. Accordingly, results deemed to be indicative.

Recently, Al-Kabi, *et al*., [28] have developed a novel approach for root detection using an extended version of Khoja stemmer. As in khoja, the algorithm in that study begins with the removal of suffixes and prefixes in the input word. However, the main difference between the two algorithms is that Khoja stemmer depends on matching the extracted stem (words after stripping off suffixes and prefixes) with patterns the in terms of their lengths, whereas in Al-kabi study the pattern is chosen according to its length and according to the common letters between the stem and the pattern. For example, given the word المنتجات (meaning: products), the algorithm removes the suffixes and prefixes at first, resulting in the stem استغفار (meaning: amnesty or forgiveness). During the matching task, threeverb patterns can be identified according to the length of that stem, these are: افتعالي, انفعاليand استفعال (transliterated as: *i-f-t-à-a-l-i, i-n-f-à-a-l-i* and*i-s-t-f-à-a-l*). However, the only pattern that have the highest number of common letters with the stem is the verb pattern استفعال (its shares four letters at positions 1, 2, 3 and 6) and thus, the pattern استفعال is chosen as the valid verb pattern for the stem منتج. As the pattern is selected, the root can be easily extracted from the matched pattern.

Results reported in Al-Kabi study showed that the proposed algorithm yields higher accuracy when it was compared to Khoja stemmer. One of the cons of the developed stemmer, however, is that it fails to extract roots from words whose lengths are less than 4 letters. In addition, the dataset that have been used in study is extremely small. It only contains 6081 Arabic words. Therefore, the results of the study can be considered as indicative rather than conclusive.

3.2.2. Light-Based Stemming and Affix Truncation

To mitigate the impact of the major drawback of root-based algorithms, which is losing stem semantics, light stemming for Arabic was also proposed. Light stemmers chop off some affixes such as plural endings in English lightly from words and without performing deep linguistic analysis. From that perspective, the majority of the approaches attempt to strip off the most frequent prefixes (*i.e.* definite articles), suffixes (*i.e.* possessive pronouns) and any antefixes or postfixes that can be attached to the beginning or endings of words. For example, light stemmers generate أعمال (meaning: works) because only prefixes (including antefixes) and suffixes (including postfixes) are removed. The decision of removing any affixes, however, is usually controlled by some heuristic rules derived from common use of these antefixes. Examples of such types of stemmers include, but are not limited to, Al-stem by Darwish and Oard [19], Aljlayl and Frieder stemmer [29], Kadri and Nie linguistic stemmer [12] and Chen and Gey stemmer [30] from California Berkeley team.

Al-stem is a light stemmer, presented by Darwish and Oard [20], which lightly chops off the following prefixes but in order from right to left (وال، فال، بال، بت، يت، لت، مت، وت، ست، نت، بم، لم، وم، كم، فم، ال، لل، في، وا، فا، لا، با) plus the following suffixes starting from right to left, too (ات، وا، ون، وه، ان، تي، ته، تم، كم، هم، هن، ها، ية، تك، نا، ين، يه، ة، هـ، ي، ا). Darwish and Oard used Al-stem in their experiment to develop a technique for Arabic-English cross-language information retrieval at TREC 2002. By the term cross-language IR, it is meant the query is written in a language that is different from documents' language. In that study, Al-Stem was compared to light8 stemmer, which will be illustrated later in this section. Results concluded that the there almost no difference statistically between the two stemmers when they were tested using TREC 2001 data. Later, Al-Stem has been modified by David Graff from the Linguistic data Consortium (LDC) to strip-off the suffixes (تا and ا) and the prefixes (سي and تت) from the list of suffixes in Al-Stem.

Based on the assumption that light stemming preserves the meaning of words, unlike root-based techniques, Aljlayl and Frieder [29] proposed an algorithm to stem Arabic words lightly. The algorithm strips the most prevalent suffixes (*i.e.* possessive pronouns), prefixes (*i.e.* definite articles), antefixes or postfixes that can be affixed to the beginning of the prefixes or the end of suffixes. Aljlayl and Frieder, however, did not list their removable sets of prefixes and suffixes explicitly. The removal of affixes, however, in Aljlayl's work had been controlled by an algorithm depending on the remaining numbers of letters in the word under stemming.

After the input word is fed to the algorithm, the stemmer truncates the letter و (pronounced as WAW and it means and) only if the length of the word is greater than or equal to 3. Following this, articles are truncated from the beginning of words. If the length is of the input word is still greater than or equal to 3, longest suffixes are removed if and only if the remaining letters are 3 or more. Next, the algorithm truncates prefixes from the produced word in the previous step, but, if and only if the remaining letters are also greater than or equal 3. The last step is repeatedly performed until the stem is obtained. In some cases the algorithm uses a normalization technique for words as well as removing all the diacritical marks except the diacritical mark shadda. This is because shadda is a sign for a duplication process of a consonant and thus it exemplifies a letter that could be lost if shadda is removed. One advantage of the algorithm is that it can deal with some arabicized words according to a predefined list. Arabicization referred to Arabic transliterated, rather than translated, words that are borrowed from other languages e.g., كمبيوتر (meaning: computer). Arabicized words in Arabic are often nouns and terminology derived from other languages. However, entries in such an arabicized list would probably be limited in its coverage. Aljlayl and

Frieder concluded that their light stemming algorithm outperforms root-based algorithms, in particular the Khoja stemmer.

Larkey, Ballesteros and Connell [31] proposed several light stemmers (light 1, light 2, light 3 and light 8) based on heuristics and some strippable prefixes and suffixes. The affixes to be removed are listed in **Table 5**. In the implementation, the algorithms of these different versions of light stemming perform the following steps:

- Peel away the letter و (meaning: and) from the beginning of words for light 2, light 3, and light 8 only if there are 3 or more remaining letters after removing the و. Such condition attempts to avoid removing words that start with the letter و.
- Truncate definite articles if this leaves 2 letters or more.
- Remove suffixes, listed in table below from right to left, from the end of words if this leaves 2 letters or more.

In monolingual and cross lingual experiments, developers of light 8 concluded that it outperforms the Khoja stemmer, especially after removing stopwords with or without query expansion. Actually, Larkey, Ballesteros and Connell concluded that removing stopwords results in a small increase in average precision, which is statistically significant for light 2 and light 8, but not for raw (the case of no stemming or normalization) and normalized words. In the same experiments, Larkey, Ballesteros and Connell used co-occurrence analysis, based on a string similarity metric, to refine some simple stemmers that are light stemmers followed by removal of vowel letters plus HAMZA (ء). From the experiment, it is concluded that a repartitioning process consisting of vowel removal followed by refinement using co-occurrence analysis performed better than no-stemming or very light stemming. In contrast, light8 stemming followed by vowel removal and the co-occurrence analysis is not better that light8 with stop word removal.

Larkey, Ballesteros and Connell [4] expanded their previous studies by adding another light stemmer called light 10. In fact, among the set of the Arabic light stemmers, the most famous, and yet the most elegant and heavily used one, is light 10 [4]. Light 10 is an extension to Larkey's light stemmers set and in particular it is the latest update of light 8 in her set. Light 10 has been identified as the best ever developed stemmer for Arabic language. In light-10, Larkey and her team proposes to lightly chops off the prefixes (ال، وال، بال، كال، فال، لل، و) from the beginning of words plus the suffixes (ها، ان، ات، ون، ين، يه، ية، هـ، ة، ي) from the end. However, the removal of affixes in the algorithm is controlled with three rules:

Table 5. Strippable strings removed in light stemming.

Light stemmer type	Removing from front	Removing from end
Light1	ال، وال، بال، كال، فال	none
Light2	ال، وال، بال، كال، فال، و	none
Light3	ال، وال، بال، كال، فال، و	هـ، ة
Light8	ال، وال، بال، كال، فال، و	ها، ان، ات، ون، ين، يه، ية، هـ، ة، ي

1) Peel away the letter و (meaning: and) from the beginning of words if there are 3 or more remaining letters after removing the و.

2) Truncate definite articles if this leaves 2 letters or more.

3) Remove suffixes, starting from right to left, from the end of words if this leaves 2 letters or more.

The robust feature of light 10 and in light stemming approaches in general, is that the stemmer minimizes the impact of the over-stemming problem. Since only few prefixes and suffixes are removed then the semantic meanings of words will be preserved. Consider the word الطفيليات. If the word is lightly stemmed, then the resulted stem is طفيل (as only the definite article prefix ال and the plural feminine suffix ات will be eliminated according to the algorithm). It is noticed that both the word and the stem have the same semantic meaning. In general, this is a very strong feature for light-stemming approaches. In the experiments, the developers of light 10 showed that it outperforms Khoja stemmer and the difference is statistically significant.

In the same study, the produced stems using light10 was also compared to the generated stems after words were processed using both Buckwalter and Diab analyzers [26] [32]. Diab Analyzer [32] is an Arabic morphological software developed to resolve the tokenization, POS tagging and Base Phrase Chunking problem of MSA. The analyzer utilized a supervised learning approach that uses training data taken from the Arabic Tree Bank and is based on using SVM (Support Vector Machines). The assumption made here is that problems like tokenization and part of speech tagging, for examples, can be considered as some types of classification problems in which the task is to predict the tag of the token's class, based on a trained number of features that are extracted from a predefined linguistic context. Thus, in the experimental setup of the experiments conducted by Lareky and her team [31], Diab analyzer was used to tag words and then according to this tagging process several runs were tested. For example, by referring to tags of words that are generated by Diab analyzer, light 10 determines either to truncate suffixes or to truncate only some of these suffixes. For instance, if the tagger tags a word as dual or plural proper nouns or plural nouns, light10 truncates only dual and plural endings from input words. In the study, results concluded that light 10 outperformed both Buckwalter and Diab analyzers and the differences are statistically significant.

In spite of the above stated conclusion about light 10, but yet the stemmer still have major drawbacks that can be identified. The obvious one is the under-stemming problem, in which words with the same meanings may be clustered into different groups. For instance, the stemmer fails to group the words اقتتل (meaning: they are fighting hardly with each others), which is stemmed to اقتتل, and القاتل (meaning: the killer), which is stemmed to قاتل, although both words are semantically similar. As a result, the stemmer may result in low recall as many relevant documents will not be retrieved. Under-stemming is limited to light 10 only and it appears in every light stemmer in Arabic studies.

Inspired by the drawbacks of both light and heavy stemming techniques, Kadri and Nie [12] proposed a new stemming technique known as linguistic-stem. The developers employed Arabic morphological rules to produce possible stems of words. The task consists of two phases. In the first phase, which is the training, 523,359 different tokens in the TREC collection were firstly extracted so as to create corpus-based statistics. The role of these statistics is to determine the most common stems because the latter can resolve easily any ambiguity occurring when words are stemmed. Accordingly, in the training phase, every word in the corpus has been decomposed according to some rules to produce all the possible stems. Thus, a corpus of stems with their occurrence frequencies was built and the most common stems are listed with their frequencies. However, the developers didn't list the rules that were applied in the study. In the second stage of the study, which is the stemming, Kadri technique truncates the most common prefixes and suffixes (as in other light stemmers) that were obtained from corpus statistics. So, if there is only one stem, the stem is returned; otherwise, the algorithm chooses the most appropriate stem depending on the pre-estimated statistics. In the same study, which was implemented using Arabic TREC collection, another light stemmer that was developed by the same author, was compared. In that light stemmer, the same argument of computing some corpus statistics but, for antefixes rather than for stems was also used. The used prefixes were (وبال, فل, ولل, كال, فال, ول, وب, بال , وال, ل, ب, ال, ا, و, فب) and the list of suffixes contains(تي, هما, وا, آ, نا, هم, ون, ات, ان, ين,و , ها, ت, ي, ن, ه, ا). Results reported in the study showed that the proposed linguistic stemmer yields better results than the used light stemmer and the difference in performance is statistically significant. However, it is noticed that the complete details for how the rules have been applied was not provided in the study.

Chen and Gey [30] from California Berkeley team follow the same technique to stem Arabic words with the ability to remove 26 prefixes and 22 suffixes. The two lists were identified manually according to grammatical roles that these affixes play in language. Additionally, corpus based statistics (occurrence of affixes frequencies), empirical evaluation for the previous experiments of TREC and English translations of the affixes.

The algorithm that handles affixes removal is well controlled in the Berkeley team work.

For the removal of prefixes, the algorithm checks the length of the input word and according to that length; a specific rule will be applied. If the length is:

- At least 5 characters, truncate the first 3 characters if and only if the first three letters of the word is in the list (وال, بال, كال, ولل, مال, اال, سال, لال).
- At least 4 characters, truncate the first 2 characters if and only if the first two letters of the word is in the list (با, لل, وم, وت, لا, سي, وس, وي, ول, كا, فا). However, if the word begins with the letter و, remove that letter.
- At least 4 characters and the initial letter in the word is ب or ل, then truncate the occurring letter if and only if the produced word (after the letter ب or ف is being truncated) is in the TREC 2002 Arabic collection.

For stripping-off suffixes, the algorithm also checks the length of the word (before removing the suffix but after the removal of the prefix). If the length is:

- At least 4 characters long, truncate recursively the following letters according to their occurrences in the list: (ها, ية, هم, وا, نا, وا,ما , يا, ني , هن, كم, كن, تم, تن, ين, ان, ات, ون).
- At least 3 characters long, truncate recursively the following one-character suffixes that appear in the list: (ة, ي,ه , ت).

The results reported by Chen and Gey study, showed that the algorithm was very beneficial to retrieval performance.

Inspired by the fact that some light stemmer may result in removing some affixes that are parts of the original words, Nwesri, *et al.*, [33] [34] proposed some heuristic algorithms to strictly and lightly strip-off prefixes, which are conjunctions and prepositions as described earlier. The developed algorithms are based on how to identify prepositions from conjunctions like ف and ل (pronounced as: FAA and LAM). The algorithms consist of several steps and they make use of some Arabic lexicons and spelling checkers with more than 15 million Arabic words. The experiments, which were tested using TREC 2001, showed that the improvement of in performance was statistically significant.

Ababneh, *et al.* [35] developed a new rule-based stemmer which firstly matches the input word with a single pre-defined list of some Arabic patterns, e.g., فاعل. This is performed firstly in the study so to take a decision whether to remove possible prefixes from words begin with them or not. For instance, consider the word كامل (meaning: the proper noun Kamil). Since the word matches the pattern فاعل then the prefix كا will not be removed as the prefix would be a part of the word. Thus, the word in which the prefix occurs would be retained as a stem. In the algorithm, if the word does not match any pattern, then the possible combinations between prefixes and suffixes in the word are examine—according to some compatibility table. If the combination is not valid then the word is preserved and returned as a stem; otherwise some heuristic rules that are based on counting the length of words after and/or before truncating affixes, are applied. The experiments reported in Ababneh and his team [35], however, were conducted with a sample term lists containing only 21 words.

Very recently, Sameer [36] developed an approach for stemming Arabic words using similar techniques to those in light stemming. The algorithm is really simple. It depends on pre-defined and ordered lists of suffixes and prefixes. During the algorithm, the developers remove occurring suffixes and prefixes according to their order in their corresponding table. The algorithm was not tested sufficiently as only 14 words were used to test the algorithm.

Morphological analyzers for dialectal Arabic have been also proposed. It is known that fro Arabic language, there is a continuum of spoken dialects varying geographically, but also by social class, which are native languages. These dialects differ phonologically, lexically, morphologically and syntactically from one another [2] [37]. Regional variation problem is one of the challenging problems to Arabic IR and NLP. Examples for such analyzers are ADAM (Analyzer for Dialectal Arabic Morphology) [38]. AbuAtta and Al-Omari [39] also attempts to produce stem for words written in Gulf dialect. As in ADAM, the authors developed a set of morphological rules to remove suffixes and prefixes from Arabic Gulf dialect texts. As in MSA stemmers, the algorithm implements also some rules for truncation that are based on word length. The results indicated that MSA stemmers were not able to extract the stems of Arabic Gulf text unless their rules are modified. Therefore, the developed stemmer yields better results than MSA stemmers. Results were tested using small collection.

3.2.3. Statistical Stemming

Motivated by the fact that the stemming technique is a language-dependent process, statistical-based stemmers that demonstrate as language-independent techniques to conflation were also proposed for Arabic IR. Examples of statistical stemmers are those based on corpus analysis [31] [40]. The basic principle made here is that since conflated words in a given corpus tend to co-occur together in the same corpus, then the relationship between words, which is usually computed using some similarity measure (*i.e.* Mutual Information or Dice Coefficient), can be utilized to prevent, for example, two semantically different words that have the same stem to be grouped together even if the stemmer produces a single stem for those conflated words. For example, the words طفيليات and أطفال should not be grouped in the same stem (or as in English the words police and policy) because making use of co-occurrence statistics would result in distinguishing words from each others.

From that perspective, using co-occurrence statistics and association relationship measures (*i.e.* Mutual Information) between word pairs, makes it possible to determine which words are semantically different and which are similar, even if the words have the same letters. For instance, consider the Arabic word. ذهب the word is a polysemous as two meanings can be provided: go and gold. Accordingly, by making use of co-occurrence statistics, the two words should be stemmed to different clusters if the context in which they appear indicates such distinctive meanings. Reported results by Xu and Croft [40] and by Larkey, *et al.* [31] showed that the approach is effective for improving stemming but, yet it was not found to be better than light-10.

The use of association similarity measures to words level has been also used for Arabic IR stemming. The premise here is that segmenting each word into a set of 3-grams for example, and computing a similarity measure, using Dice Coefficient for example, between the set of the 3-grams of that word and the set of the possible 3-grams of the query word would result in a similarity value that might allow clustering the word in a specific class. The major advantage for the use of N-grams models is that they are able to capture broken plurals. In spite of that broken plurals do not get conflated with their singular forms because they preserve some affixes and internal differences, but, yet, the singular form and broken plural of a certain word have some common letters in many broken plurals. Accordingly, segmenting the word in its singular and plural forms would probably capture the shared letters and hence, the plural and also different inflected forms of words are thus, clustered.

Using these arguments, Mustafa and Al-Radaideh [41] stated that the use of a di-gram method for Arabic IR offers better performance than tri-grams with respect to precision and recall ratio but, the method is not an effective solution to corpus-based Arabic word conflation because the language richness.

The same technique had been also used by Khreisat [42], who used both Dice similarity and Manhattan dissimilarity coefficients. The contribution of this study is that it showed that the use of similarity measures is much better than using dissimilarity coefficients. In the study the used N-grams was the tri-grams. The total number of the used documents in Khreisat was not provided in the study.

Hmeidi, *et al.*, [43] followed a similar approach to Khreisat. They used the same similarity and dissimilarity measures, which are Dice and Manhattan coefficients. They concluded that the use of bi-grams Arabic tokens is efficient to extract Arabic roots regardless of their length (*i.e.* trilateral or quadliteral roots). The algorithm was tested with only 242 abstract plus the texts of the Holy Quran. As in Khreisat work, the results concluded that the use of similarity measure yield better results than the use of dissimilar measure.

Xu, *et al.* [22] combined Arabic monolingual N-gram retrieval with stemmed words. The study showed that the use of tri-grams combined with stemming, improved retrieval, though this improvement is not statistically significant. The study also experimented with bi-grams and di-grams, instead of tri-grams. Results indicated that both of them do not outperform tri-grams because bi-grams are very short with little context while di-grams are similar to word or stem-based retrieval. In the study, the use of the N-grams was able to detect broken plurals.

The same authors extended their study to include spelling normalization [22] In that study, spelling normalisation (variants in spelling) was implemented to detect the confused cases of some letters (*i.e.* YAA and ALIF). In the experiments, Xu and his colleagues concluded that the use of spelling normalisation for orthographic variation with 3-grams and stemming improves Arabic retrieval performance significantly by 40%. Surprisingly, in this experiment, Xu and his colleagues stated that stemming and spelling normalisation have a small impact on cross language information retrieval, unlike the results by the developers of the light10 stemmer, who used the same TREC 2001 data. With respect to stemming, Larkey, *et al.*, [19] explained that Xu, *et al.*, [44] used a parallel corpus, extracted from a UN corpus, so their bilingual lexicon contains several variants of Arabic words. However, Larkey, Ballesteros and Connell used a bilingual lexicon derived from an online dictionary, so it contains fewer variants. This means that query terms were not matched against the dictionary entries unless they were stemmed.

Chen and Gey [30] implemented a new approach for Arabic stemming using statistical stemmers that make use of parallel corpora (several monolingual corpora translated in more than one language). Chen and Gey used an English stemmer to stem English words in an English-Arabic parallel corpus. Then, Arabic words are clustered together into a stem category depending on their mappings/translations to English stems in the corpus after being aligned and processed with GIZA++ [45], which is a statistical MT toolkit that was designed for alignment in parallel corpora. Results showed that the increase in performance was substantial when it was compared with Al-stem.

3.2.4. Simple Tagging Based Stemming

Stemming based on light/simple tagging has been also utilized for Arabic texts. The idea is to lightly tag words into some different tags so as to use different types of stemming techniques. Al-Shammari and Lin [46] used a list of 2200 stopwords to classify verbs from nouns with a hypothesis that some stopwords precede nouns, e.g., من (meaning: from or to) whereas others precede verbs e.g., لم (لم is an Arabic conjunction means never and is used for the negation of the verb that follows). In the study, the Khoja stemmer was applied to stem verbs while light stemming is applied to nouns. Using two samples of data, in particular 47 medical documents with 9435 words and 10 sports articles (7071 words), Al-Shammari and Lin evaluated their stemmer, which was called Educated Text Stemmer (ETS) and they concluded that their stemmer was able to generate 96% correct stems. In addition, they stated that the ETS stemmer produces better results when more documents are contained in the stemming process.

Mansour *et al.* [44] presented an auto-indexing approach to build indices for Arabic documents. In their indexing process, the algorithm firstly tagged every word into verbs and nouns using morphological rules. The process was managed by a set of predefined rhythms (patterns). Secondly, the algorithm removes stopword and stop-list phrases. Thirdly, the algorithm identifies nouns and verbs depending on the preceding word, as it illustrated in the stemming section in this chapter. Fourthly, the algorithm extracts stems from the rhymed/patterned words. In particular, some morphological rules were used to extract stems from both nouns and verbs. For instance, verbs were checked firstly against some exceptional grammatical rules for Arabic verbs. If such scenario fails, then verbs are checked against the "ten-verb-additions" rule (grammarians of Arabic stated that the derivative system of any verb has 10 known different formats) after being heavily investigated to remove non-essential letters and thus the stem of any verb is obtained. Finally, Mansour and his colleagues assign weights to the stemmed words relative to their documents, depending on some statistical factors like the frequency of occurrence of a word in its containing document. Thus, all the possible stems of a word will be sorted according to their weights. Developers concluded that their method is very useful and obtain an average recall of 46% and an average precision of 64% when it is tested with 24 arbitrarily selected general-purpose texts with various lengths.

Both methods of Mansour and Al-Shammari seem reasonable but the experiments were not comprehensive as small and non-standard sets were used (only 24 texts were used by Mansour).

3.2.5. Artificial Intelligence Approaches to Stemming

Inspired by the fact that Neural Networks (NN) can be applied to a large number of applications, Alserhan and Ayesh [47], proposed the use of Back-Propagation Neural Network (BPNN) for stemming Arabic text. The work was motivated by the fact that the majority of the current root-based approaches employed relatively large morphological rules, which could have a real impact on storage and time required to access. The classification is based on the frequent appearance of the letters in Arabic language. So, in the study, Alserhan and Ayesh classified Arabic letters into four different classes that represent Arabic affixes. Each class is assigned a value of a three binary digits. During training stage, the neural network was trained using 250 words with their correct roots. Results reported by the study, showed that the use of neural networks in Arabic stemming could yield the correct root with an accuracy of 84%. The major advantage of Alserhan and Ayesh study is that it does not depend on any morphological rules but, yet, the study has two major drawbacks. First, it was limited to Arabic words with a maximum of length 4. Second, the results were only tested by 1000 words.

Boubas, *et al.*, [48] proposed the use of Genetic Algorithms (GA) to handle problem of Arabic stemming. The population in the study is built with 3089 solutions, which represent all possibilities that can be produced from a single root after being combined with suffixes and prefixes. However, during building the morphological system, only tri-laterals were used and thus, only addition and removal of prefixes and suffixes are applied, but not their substations or replacements. The possible combinations, with their examples, were fed to the system, in terms of learning patterns, during learning phase so as to generate morphological rules for verb patterns. During root-extraction phase, the system performs string matching so as to match input word with the morphologically extracted rules. The authors claimed that their machine learning system is able to produce roots from any Arabic word. In spite of the novelty of the proposed approach, the developers didn't prove how the system is tested. In addition, they didn't illustrate how the complete system works (*i.e.* fitness function employed).

4. Discussion

It is apparent that in highly morphological languages such as Arabic stemming could have a significant impact on retrieval. This is very evident in the majority of the studies provided in the paper. It is also clear that heavy and light stemming approaches are the most dominant ones among the existing approaches for stemming Arabic but, it can be concluded from the reported studies that light-based stemming is better than heavy-based stemming. But, each of the two paradigms has some pros and cons. On one hand, heavy stemming often results in over-stemming, leading to a low precision. This is especially true in morphologically rich languages, as Arabic, which are often rich of polysemous words in which a single word could have multiple meanings. So, rendering infected forms of words into a single root would probably results in returning large number of irrelevant documents.

On the other hand, light stemming may not succeed to cluster semantically similar words together (under-stemming), resulting in low recall. However, in spite of this major drawback, light 10 is the best known algorithm for indexing Arabic texts. It has been identified as a fashionable solution to Arabic stemming problem. Therefore, light 10 has been added to the most famous IR systems like the Lucene and the Lemur toolkit.

It is true that light stemming preserves the meaning of words, unlike root-based techniques, and achieves the goal of retrieving the most pertinent documents, but in the opinion of the authors of this paper, the major reason behind the success of light-based stemming over root-based stemming is that the strippable affixes in the former approach are those appearing in Arabic nouns and adjectives (*i.e.*, وال، بال، كال، فال,) and the belief is in Arabic nouns and adjectives are much larger than verbs. In fact, there are only few morphological rules, known as "ten-verb-addition", to formulate verbs from roots. In contrast, root-based stemming techniques, which often attempt to produce root, focus on verbs and handle even nouns and adjective in the same context. It is evident that it is not always correct to produce the root of proper nouns or nouns in general. Let's consider the following nouns: السعودية, المكاني, المهرجان, باراك أوباما and الستائر (meanings respectively: Saudia, the festival, spatial, the US leader Barak Obama). Using a root based stemmer like Khoja, the stems are سعد, كني, هرج, برا وبا and ستر, respectively. All of the stems are either chaotic or/and do not have similar semantic meanings to their original words. These two facts are the main causes for why light stemming techniques outperform root-based st.

However, in IR the most semantically clustered words, the better retrieval task. Therefore, in spite of the achievements of light stemming techniques, in general (and light 10, in particular), the belief is that they are not the best paradigm for indexing Arabic texts. In addition, it is obvious that in a rich language like Arabic, the process of relatively blind removal of affixes (*i.e.* prefixes and suffixes) could have a significant impact on words and may lead to mis-stemming and ambiguity problems, in which some letters that are original in words are erroneously stripped-off. In fact, light stemming techniques are really simple as they depend solely on the removal of affixed and some controlled rules devised experimentally.

When it comes to Arabic English CLIR, in which the query language is different from the language that presents in document collection, the belief is that it may result in some relatively high OOV (Out-of-Vocabulary) words. This refers to that the majority of the Arabic-to-English dictionaries (not the opposite) list their entries in terms of roots. In fact, whenever Arabic native speakers need to translate an Arabic word into English, they always render Arabic words to their roots to increase the possibility of capturing the translation senses. This is an accredited point for root-based techniques.

In the opinion of the authors, the only way to avoid ambiguity that may occur due to blind removal of affixes (when a letter or some consecutive letters are not parts from words), is to use some statistical data extracted from corpora. It is very evident (as in Kadri and Nie study above) that such corpus statistics are very useful for handle such ambiguity because the removal decision of affixes depends on the distribution of that affixes in the corpus and whether it is most frequent or not and thus, our certainty about the removal process is handled and estimated, which could help a lot in the final decision.

Nevertheless, the use of corpus statistics just imposes burden of removal ambiguity to domain and size of that certain corpus. The belief of the authors is that there is always a possibility of undesirable behavior and/or poor performance once moving from one domain to another domain and/or when the corpus size changes. This is not surprising because using statistics depends solely on the size and the domain from which data is sampled. Consider, for example, Arabic technical words in computer science. There are a considerable number of words that are borrowed from other languages (*i.e.* English). So, using of corpus statistics to avoid removal ambiguity, with a corpus of computer science may result in dropping performance of some technique radically because news collections, for examples, may have unique features that may not be found in other genres.

It is also noticed in the reported studies in Arabic stemming, that a considerable number of the developed approaches had been tested using small collections, rather than using standard corpora (*i.e.* TREC 2001 and 2002). This is a major drawback in the developed approaches because their results deemed to indicative rather than conclusive in this case. It is not guaranteed to get the same achieved results, when such techniques are tested using standard test beds.

The majority of the developed techniques didn't show how they handle broken plurals in Arabic. In fact, only few studies addresses the problem in terms of statistical stemming and/or using some clustering approaches or translation techniques using an aligned corpora as it has been described earlier in this paper. As illustrated earlier, broken plurals represent 10% of Arabic texts. It is not avoidable and the belief is that the problem should be handles within stemming techniques.

The use of simple part of speech tagging, statistical stemming and artificial intelligence are very useful to Arabic stemming task. Statistical stemming (*i.e.* N-grams models) and artificial intelligence techniques, for examples, have the ability to detect broken plurals of Arabic words, unlike root and heavy based techniques. They also have the ability of cluster words that are related together and to minimize polysemy impact (*i.e.* when a word have multiple meanings, clustering could distinguish between these meanings). Nevertheless, the use of such techniques increases performance penalty needed to identify clusters and/or detect tag-of-speech of words. In addition, simple POS tagging techniques rely on a hypothesis that does not always holds, which is the preceding words. In fact, the majority of the Arabic words cannot be determined by only preceded words. This is may be the major reason for using only small text collections for reported experiments which used such technique.

Orthographic variation in Arabic and various writing of some letters could also have a significant, because incorrect normalization may yield a stemmed word that does not share the meaning with the original word. Such orthographical differences should be handled carefully within stemming techniques. In the belief of the authors of this paper, Arabic stemming task should not be dependent on a specific approach. This is the only method to develop an accurate and error-free Arabic stemmer.

5. Conclusions

Arabic language is an extremely rich with its morphology, derivational system and grammatical rules. For such a language, stemming techniques could have a significant impact on improving retrieval performance. This paper reviews a considerable number of studies that have been conducted to resolve Arabic stemming problem. Several studies are presented and the causes for why Arabic language is challenging and its implications on NLP and IR have been well analyzed in the paper.

However, in spite of the achievements, it is yet not apparent which approach is the best for indexing Arabic texts. It is true that in NLP and IR research, community light stemming techniques have been widely adopted for their simplicity and their ability to preserve words meanings. But, yet, they are still far from the optimal accuracy. Root-based stemmers may result in higher recall but, many irrelevant documents may be retrieved because they cluster words in different classes. Additionally, light stemmers focus on nouns and hence, they perform relatively poor for nouns. Morphological analyzers, as Khoja stemmer which firstly tokenizes the input text, could result in incorrect tokenization and in stemming consequently. Additionally, they have been adopted to focus on verbs rather than nouns. Tagging techniques could improve performance but in an ad-hoc task like IR they are not the optimum. Statistical techniques could contribute to resolving broken plural problem but, they depend solely on corpus statistics, which could be changed as we move from a specific domain to another. We can conclude that there is no optimal solution yet for the problem of how to index Arabic terms.

References

[1] Manning, C.D., Raghavan, P. and Schutze, H. (2008) Introduction to Information Retrieval. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9780511809071

[2] Mustafa (2013) Mixed-Language Arabic-English Information Retrieval. PhD Thesis, University of Cape Town, Cape Town.

[3] Darwish, K. and Magdy, W. (2014) Arabic Information Retrieval. *Foundations and Trends in Information Retrieval*, **7**, 239-342. https://doi.org/10.1561/1500000031

[4] Larkey, L., Ballesteros, L. and Connell, M. (2007) Light Stemming for Arabic Information Retrieval. In: Soudi, A., van den Bosch, A. and Neumann, G., Eds., *Arabic Computational Morphology*, Springer, Berlin, 221-243. https://doi.org/10.1007/978-1-4020-6046-5_12

[5] Pirkola, A., Hedlund, T., Keskustalo, H. and Järvelin, K. (2001) Dictionary-Based Cross-Language Information Retrieval: Problems, Methods, and Research Findings.

Information Retrieval, **4**, 209-230. https://doi.org/10.1023/A:1011994105352

[6] Mirkin, B. (2010) Population Levels, Trends and Policies in the Arab Region: Challenges and Opportunities. Arab Human Development, Report Paper 1.

[7] Cheung, W. (2008) Web Searching in a Multilingual World. *Communications of the ACM*, **51**, 32-40. https://doi.org/10.1145/1342327.1342335

[8] Habash, N. and Rambow, O. (2007) Arabic Diacritization through Full Morphological Tagging. *Human Language Technologies*: *The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, 22-27 April 2007, 53-56.

[9] Manzour, I. (2017) Lisan Al-Arab. www.lesanarab.com

[10] Hegazi, N. and El-sharkawi, A. (1985) An Approach to a Computerized Lexical Analyzer for Natural Arabic Text. Proceedings of the Arabic Language Conference, Kuwait, 14-16 April 1985.

[11] Mustafa, M. and Suleman, H. (2011) Building a Multilingual and Mixed Arabic-English Collection. 3*rd Arabic Language Technology International Conference*, Alexandria, 17-18 July 2011, 28-37.

[12] Kadri, Y. and Nie, J.Y. (2006) Effective Stemming for Arabic Information Retrieval. *Proceedings of the Challenge of Arabic for NLP*/*MT Conference*, London, 23 October 2006, 68-74.

[13] Attia, M.A. (2008) Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation. PhD Thesis, The University of Manchester, Manchester.

[14] Attia, M.A. (2007) Arabic Tokenization System. *Proceedings of the* 2007 *Workshop on Computational Approaches to Semitic Languages*: *Common Issues and Resources*, Prague, 28 June 2007, 65-72. https://doi.org/10.3115/1654576.1654588

[15] Goweder, A., Poesio, M., De Roeck, A. and Reynolds, J. (2005) Identifying Broken Plurals in Unvowelised Arabic Text. *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, Vancouver, 6-8 October 2005, 246-253.

[16] Buckwalter, T. (2004) Issues in Arabic Orthography and Morphology Analysis. *Proceedings of the Workshop on Computational Approaches to Arabic Script-Based Languages*, Geneva, 28 August 2004, 31-34. https://doi.org/10.3115/1621804.1621813

[17] Levow, G.A., Oard, D.W. and Resnik, P. (2005) Dictionary-Based Techniques for Cross-Language Information Retrieval. *Information Processing & Management*, **41**, 523-547. https://doi.org/10.1016/j.ipm.2004.06.012

[18] Abdelali, A. (2006) Improving Arabic Information Retrieval Using Local Variations in Modern Standard Arabic. PhD Thesis, New Mexico Institute of Mining and Technology, New Mexico.

[19] Darwish, K. and Oard, D.W. (2003) CLIR Experiments at Maryland for TREC-2002: Evidence Combination for Arabic-English Retrieval.

[20] Daoud, D. and Hasan, Q. (2011) Stemming Arabic Using Longest-Match and Dynamic Normalization. 3*rd Arabic Language Technology International Conference*, Alexandria, 17-18 July 2011, 54-59.

[21] Xu, J., Fraser, A. and Weischedel, R. (2001) TREC 2001 Cross-Lingual Retrieval at BBN. *Text Retrieval Conference*, Gaithersburg, 13-16 November 2001, 68-78.

[22] Xu, J., Fraser, A. and Weischedel, R. (2002) Empirical Studies in Strategies for Arabic Retrieval. *Proceedings of the* 25*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, 11-15 August

2002, 269-274. https://doi.org/10.1145/564376.564424

[23] Khoja, S. and Garside, R. (1999) Stemming Arabic Text. Computing Department, Lancaster University, Lancaster.

[24] Hammo, B.H. (2009) Towards Enhancing Retrieval Effectiveness of Search Engines for Diacritisized Arabic Documents. *Information Retrieval*, **12**, 300-323. https://doi.org/10.1007/s10791-008-9081-9

[25] Darwish, K. (2002) Building a Shallow Arabic Morphological Analyzer in One Day. *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, Morristown, July 2002, 1-8. https://doi.org/10.3115/1118637.1118643

[26] Buckwalter, T. (2002) Buckwalter Arabic Morphological Analyzer Version 1.0. Linguistic Data Consortium, University of Pennsylvania, Philadelphia.

[27] Ghwanmeh, S., Kanaan, G., Al-Shalabi, R. and Alrababah, S. (2009) Enhanced Algorithm for Extracting the Root of Arabic Words. 6*th International Conference on Computer Graphics*, *Imaging and Visualization*, Tian Jin, 11-14 August 2009, 388-391.

[28] Al-Kabi, M., Kazakzeh, S., Abu Ata, B., Al-Rababah, A.S. and Izzat, A.M. (2015) A Novel Root Based Arabic Stemmer. *Journal of King Saud University—Computer and Information Sciences*, **27**, 94-103. https://doi.org/10.1016/j.jksuci.2014.04.001

[29] Aljlayl, M. and Frieder, O. (2002) On Arabic Search: Improving the Retrieval Effectiveness via Light Stemming Approach. *Proceedings of the* 11*th ACM International Conference on Information and Knowledge Management*, Illinois, 4-9 November 2002, 340-347. https://doi.org/10.1145/584792.584848

[30] Chen, A. and Gey, F. (2002) Building an Arabic Stemmer for Information Retrieval. *Text Retrieval Conference*, Gaithersburg, 19-22 November 2002, 631-639.

[31] Larkey, L.S., Ballesteros, L. and Connell, M.E. (2002) Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-Occurrence Analysis. In: *Proceedings of the* 25*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, 11-15 August 2002, 275-282. https://doi.org/10.1145/564376.564425

[32] Diab, M., Hacioglu, K. and Jurafsky, D. (2004) Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Boston, 2-7 May 2004, 149-152. https://doi.org/10.3115/1613984.1614022

[33] Nwesri, A.F.A., Tahaghoghi, S.M.M. and Scholer, F. (2005) Stemming Arabic Conjunctions and Prepositions. *Lecture Notes in Computer Science*, **3772**, 206-217. https://doi.org/10.1007/11575832_23

[34] Nwesri, A., Tahaghoghi, S.M.M. and Scholer, F. (2007) Arabic Text Processing for Indexing and Retrieval. *Proceedings of the International Colloquium on Arabic Language Processing*, Rabat, 18-19 June 2007, 18-19.

[35] Ababneh, M., Al-Shalabi, R., Kanaan, G. and Al-Nobani, A. (2012) Building an Effective Rule-Based Light Stemmer for Arabic Language to Improve Search Effectiveness. *International Arab Journal of Information Technology*, **9**, 368-372.

[36] Sameer, R. (2016) Modified Light Stemming Algorithm for Arabic Language. *Iraqi Journal of Science*, **57**, 507-513.

[37] Habash, N. and Rambow, O. (2006) MAGEAD: A Morphological Analyzer and Generator for the Arabic Dialects. *Proceedings of the* 21*st International Conference on Computational Linguistics and the* 44*th Annual Meeting of the Association for Computational Linguistics*, Sydney, 17-21 July 2006, 681-688.

https://doi.org/10.3115/1220175.1220261

[38] Salloum, W. and Habash, N. (2014) ADAM: Analyzer for Dialectal Arabic Morphology. *Journal of King Saud University—Computer and Information Sciences*, **26**, 372-378. https://doi.org/10.1016/j.jksuci.2014.06.010

[39] Abu Ata, B. and Al-Omari, A. (2014) A Rule-Based Stemmer for Arabic Gulf Dialect. *Journal of King Saud University—Computer and Information Sciences*, **27**, 104-112.

[40] Xu, J. and Croft, W.B. (1998) Corpus-Based Stemming Using Cooccurrence of Word Variants. *ACM Transactions on Information Systems*, **16**, 61-81. https://doi.org/10.1145/267954.267957

[41] Mustafa, S.H. and Al-Radaideh, Q.A. (2004) Using N-Grams for Arabic Text Searching. *Journal of the American Society for Information Science and Technology*, **55**, 1002-1007. https://doi.org/10.1002/asi.20051

[42] Khreisat, L. (2006) Arabic Text Classification Using N-Gram Frequency Statistics a Comparative Study. *Proceedings of the* 2006 *International Conference on Data Mining*, Las Vegas, 26-29 June 2006, 78-82.

[43] Hmeidi, I.I., Al-Shalabi, R.F., Al-Taani, A.T., Najadat, H. and Al-Hazaimeh, S.A. (2010) A Novel Approach to the Extraction of Roots from Arabic Words Using Bigrams. *Journal of the Association for Information Science and Technology*, **61**, 583-591.

[44] Mansour, N., Haraty, R.A., Daher, W. and Houri, M. (2008) An Auto-Indexing Method for Arabic Text. *Information Processing & Management*, **44**, 1538-1545. https://doi.org/10.1016/j.ipm.2007.12.007

[45] Och, F.J. and Ney, H. (2003) A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, **29**, 19-51. https://doi.org/10.1162/089120103321337421

[46] Al-shammari, E.T. and Lin, J. (2008) Towards an Error-Free Arabic Stemming. *Proceedings of the* 2*nd ACM Workshop on Improving Non English Web Searching*, Napa Valley, 26-30 October 2008, 9-16. https://doi.org/10.1145/1460027.1460030

[47] Al-Serhan, H. and Ayesh, A. (2006) A Triliteral Word Roots Extraction Using Neural Network for Arabic. *International Conference on Computer Engineering and Systems*, Cairo, 5-7 November 2006, 436-440. https://doi.org/10.1109/icces.2006.320487

[48] Boubas, A., Leena, L.L., Belkhouche, B. and Harous, S. (2011) GENESTEM: A Novel Approach for an Arabic Stemmer Using Genetic Algorithms. *International Conferen-ce on Innovations in Information Technology*, Abu Dhabi, 25-27 April 2011, 77-82. https://doi.org/10.1109/innovations.2011.5893872