Preprocessing Techniques for Clustering Arabic Text: Challenges and Future Directions

Tahani Almutairi, Shireen Saifuddin, Reem Alotaibi, Shahendah Sarhan, Sarah Nassif
Department of Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

***Abstract*—Arabic is a complex language for text analysis because of its orthographic features, rich synonyms, and semantic style. Thus, Arabic text must be prepared more carefully in the preprocessing stage for the analyzer to improve the quality of the results. Moreover, many preprocessing steps have been proposed to improve the text analyzer quality by reducing high dimensionality, selecting the proper features to describe the text, and enhancing the process speed. This paper deeply investigates and summarizes the use of Arabic preprocessing techniques in Arabic text in general and focuses in-depth on clustering. Moreover, it focuses on seven preprocesses that are now used to prepare Arabic and provides the available tools for each of them; the seven preprocess are tokenization, normalization, stopword removal, stemming, vectorization, lemmatization, and feature selection. In addition, this paper investigates any work that uses synonyms and semantic techniques for preprocessing to prepare the text or reduce the dimensionality of the clustering algorithm. Therefore, this survey investigated nine techniques for Arabic text preprocessing to identify the challenges in this area. Finally, this study aims to serve as a reference for researchers interested in this area, and ends with potential future research directions.**

***Keywords*—Arabic preprocessing; Arabic language; survey; clustering; Arabic analysis**

I. Introduction

Arabic is the fourth most spoken language by native speakers, and the fifth most widely used language [1], [2], [3]. Arabic differs from other languages in that it is written and read from right to left with specific grammar, spelling, vocabulary, punctuation marks, and no case-sensitive letters. Moreover, sentences consist of verbs, subjects, and objects [4]. Arabic spoken by Arabs is known as classical Arabic (CA). Over time, CA progressed and developed into modern standard Arabic (MSA). Both versions have the same morphology and syntax but disagree grammatically and stylistically. Arabic dialects developed over time, adding to the diversity of Arabic. Dialects are informal versions of the language spoken by friends and family. Moreover, the dialects differ among Arabic countries [4], [3].

Arabic has recently attracted the attention of researchers, using various technologies and techniques. Over the past decade, Arabic and its dialects have gained attention in natural language processing (NLP) research [5], [6] , which has helped the preprocessing stage in dealing with text in different applications that benefit diverse areas, especially text analysis, such as text classification, clustering, and sentiment analysis [7]. Text clustering is an unsupervised learning technique that discovers and groups text or documents into clusters. Document clustering groups similar documents without prior knowledge of their labels. Thus, every cluster formed contains similar texts or documents. Clustering techniques identify and group texts or documents into clusters. Clustering is significant in many applications such as document retrieval, summarization, recommendation, marketing, customer analysis, document clustering, output detection, agriculture, pharmacy, and image processing [1], [8].

Many studies have used text preprocessing techniques to handle the high dimensionality of data before clustering to obtain good text clustering results [9]. Text preprocessing techniques are typically used to dramatically decrease the document size, facilitate feature selection, and enhance processing speeds. Arabic text pre-processing does not involve straightforward steps. Arabic text requires a morphological analysis that adds more challenges but is required for many reasons. The first is common orthographic variations, in which words are derived from a trilateral or quadrilateral origin system. The original system often obscures Arabic terms. Another reason for this is that Arabic synonyms within a language are extensive [1], [10], [6]. Moreover, preprocessing algorithms and techniques are limited and require further research [11].

Several Arabic preprocessing techniques, similar to other languages, have been proposed. However, some are specific and related to Arabic NLP techniques, such as tokenization, stemming, normalization, stopword removal, and lemmatization. Most Arabic preprocessing surveys focus on the classification, categorization, or specific types of Arabic fields. Only one study [12] focused on clustering to demonstrate the significant impact of term turning with stemming preprocess when vectorizing text based on TF-IDF. The authors examined and compared different text-preprocessing techniques for clustering Arabic documents. It investigated the effectiveness of techniques such as term pruning, term weighting using TF-IDF, and morphological analysis methods, including root-based stemming, light stemming, and raw text normalization. Furthermore, this study evaluates the impact of clustering algorithms, including the widely used partitional algorithm.

In contrast, [5] comprehensively classified works on three Arabic varieties: MSA, CA, and Dialect, focusing on Arabic and Arabizi types. The authors endeavored to associate each work with publicly available resources, wherever possible, to facilitate further research and development in this field. In another study [13], the authors discussed the challenges posed by Arabic in the field of NLP and provided a brief history of Arabic NLP. The tools and resources available for Arabic NLP can be broadly classified into enabling technologies that are not user-facing, and advanced user-targeting applications.

In addition, [10] proposed the extraction of information from Arabic social media text by addressing data collection, cleaning, enrichment, and availability challenges. The objective is to enhance information quality and reliability, contributing to a more effective and accurate analysis of Arabic social media content. Finally, according to [14], preprocessing is required to improve the accuracy and efficiency of Arabic information. Therefore, the author analyzed the existing techniques and identified the limitations and challenges of both types. They emphasized the importance of stemming and the need for further research to improve Arabic information retrieval.

This work explores and summarizes the application of Arabic preprocessing methods for clustering. Only one survey paper in the literature examines preprocessing methods for clustering Arabic text; other survey papers address different tasks or focus on one or two steps of the preprocessing process. This paper focus on tokenization, Arabic normalization, stop word removal, stemming, and lemmatization. It also discusses feature selection and vectorization as preparation steps for dimensionality reduction. In addition, it investigates Arabic synonyms and semantic solutions to determine whether any researcher used them as a preprocessing step to reduce the dimensionality of the clustered data.

In addition, this survey can also serve as a resource for scholars who are generally interested in Arabic preprocessing in general by answering the following questions: which preprocessing steps are used for clustering in particular, and to the Arabic language in general? Which tools are available for each preprocessing step's algorithms and techniques? Furthermore, can preprocessing be utilized to minimize dimensions and does it address the richness of synonyms in Arabic? And last, is one of the main features of Arabic that preprocessing addresses semantics?

To the best of our knowledge, this survey is one of the few that thoroughly examines clustering-focused Arabic preprocessing methods. The several preprocessing phases are covered in this paper, which also explores the field by offering resources for each Arabic language preprocessing step. While it primarily focuses on text clustering, it will also produce scientific information about Arabic preprocessing that can serve as a foundation for other domains.

The remainder of this paper is organized as follows. Section II presents the methodology used for this survey. Section III presents the Arabic preprocessing techniques. Recent studies on clustering preprocessing techniques are discussed in Section IV. Finally, the discussion and conclusions are presented in Sections V and VI, respectively.

II. Methodology

This survey was based on the PRISMA framework, a standard research strategy that searches online databases. PRISMA framework is used to systematically and carefully select high-quality and dependable references. The framework is shown in Fig. 1 and consists of four phases: identification of the research and search process, paper selection process, quality assessment, and extraction and synthesis phases.

In the first phase, the research process focuses on identifying and searching for high-quality resources that may be beneficial. This survey conducted an efficient search of libraries and journal databases, including IEEE Xplore, Springer, and ScienceDirect, to find high-quality resources. Books, survey papers, and articles that contained important information for the survey were found via Google Scholar. Also, this survey looked up pertinent articles and references using a variety of keywords. Following their collection, the documents were sorted according to a set of criteria in order to identify the most pertinent and appropriate ones.

The following criteria were used to choose and assess the research publications in the paper selection process and quality assessment phases. Initially, between 2017 and 2024, possibly pertinent papers on Arabic preprocessing had to be presented at prestigious conferences or in scholarly journals. Additionally, the study needed to be relevant to the scope of this survey's Arabic preprocessing processes, with an emphasis on or discussion of the preprocessing tool that yields excellent results for Arabic language preparation. The titles and abstracts of the gathered papers were checked to make sure they fit the survey's scope. Lastly, the remaining studies were carefully reviewed. Only the most appropriate and high-quality papers valuable to this survey were used as references in the extraction and synthesis phase, resulting in 60 research papers.

III. Arabic Preprocessing Techniques

This paper outlines the main Arabic preprocessing techniques shown in Fig. 2. The order of steps may vary depending on the model and research goals. The following subsection discusses these techniques and suggests helpful resources for each step.

A. Tokenization

Tokenization is an essential preprocessing step for any language because it is the first step that splits the long text into a small part called a "token," making it easy to use in text analysis algorithms [15], [16]. These words were then separated into numerous delimiters, including white spaces, tabs, and punctuation marks [15]. Although there are multiple tokenization algorithms, there is only one possible way to split a dataset into words. Moreover, a sub-word tokenizer allows splitting a dataset into units called "sub-words" [16], [12]. Therefore, depending on the depth of linguistic analysis, different Arabic tokenizer levels can be developed.

Because tokenization is essential, many research publications have discussed and used this technique. Similar to the authors in [15], the three tokenizers included stochastic, disjoint-letter, and morphological. The authors also discussed three other tokenizers: characters, words, and sentences. After evaluating these six tokenizers through applied unsupervised and supervised approaches, the authors provided the benefits and drawbacks of each Arabic tokenizer technique based on studying the effects of various tokenizers in Arabic for diverse Arabic classification techniques. Moreover, they created a tokenizer framework as an open-source library [1].

In addition, [10], [17] studied many Arabic preprocessing techniques on social media, including tokenization. The authors of [10] provided a combined solution for Arabic social media text preprocessing challenges at different stages: data collection, cleaning, enrichment, and availability. The proposed

[1]https://github.com/ARBML/tkseem

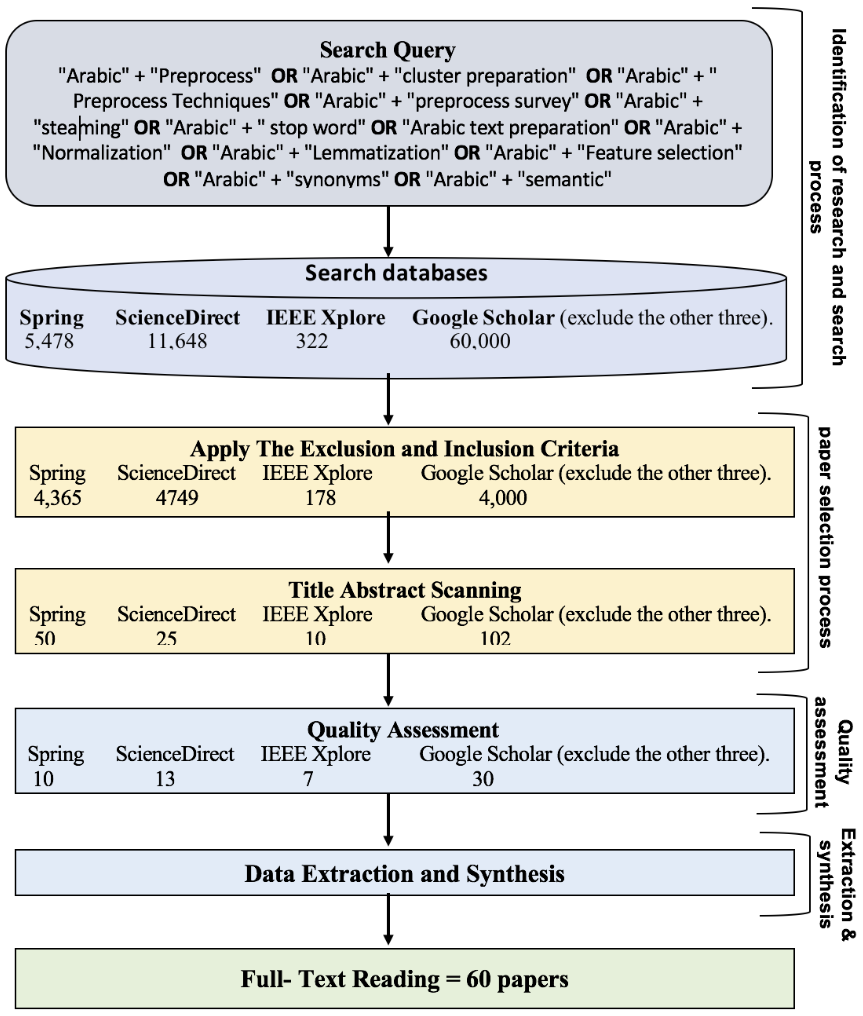
Search Query
"Arabic" + "Preprocess" OR "Arabic" + "cluster preparation" OR "Arabic" + " Preprocess Techniques" OR "Arabic" + "preprocess survey" OR "Arabic" + "steaming" OR "Arabic" + " stop word" OR "Arabic text preparation" OR "Arabic" + "Normalization" OR "Arabic" + "Lemmatization" OR "Arabic" + "Feature selection" OR "Arabic" + "synonyms" OR "Arabic" + "semantic"
Search databases
Spring 5,478
ScienceDirect 11,648
IEEE Xplore 322
Google Scholar (exclude the other three). 60,000
Apply The Exclusion and Inclusion Criteria
Spring 4,365
ScienceDirect 4749
IEEE Xplore 178
Google Scholar (exclude the other three). 4,000
Title Abstract Scanning
Spring 50
ScienceDirect 25
IEEE Xplore 10
Google Scholar (exclude the other three). 102
Quality Assessment
Spring 10
ScienceDirect 13
IEEE Xplore 7
Google Scholar (exclude the other three). 30
Data Extraction and Synthesis
Full- Text Reading = 60 papers
Identification of research and search process
paper selection process
Quality assessment
Extraction & synthesis

Fig. 1. The PRISMA flow diagram.

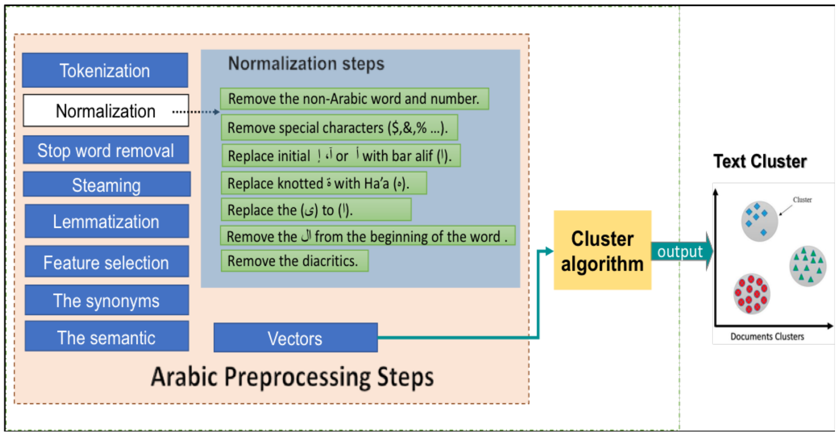

Fig. 2. Main Arabic preprocessing techniques discussed in this survey for clustering.

approach applies several NLP tools, including tokenization, stemming, and morphological analyses. The authors in [17] also investigated the impact of the 26 preprocesses applied to Arabic tweets by training a classifier to identify health-related tweets, studying the tokenization that divides the text into specific units with other preprocessing steps, where the text is divided into units in the tokenization process, and typically, those units are words.

Moreover, [12] and [18] studied Arabic document preprocessing. The authors of [12] examined and compared text preprocessing techniques in Arabic document clustering to study their effectiveness. However, they first prepared Arabic document text by tokenizing it based on words and removing stop words. In addition, [18]investigated the impact of stemming methods, an essential preprocessing step in classification. The main preprocessing steps used by the authors of [18] for Arabic include tokenization, normalization, stopword removal, feature extraction, and stemming. White space was used to tokenize the Arabic dataset.

Finally, tokenization is an essential Arabic preprocessing step because it splits long texts into small parts that are easy to use with any text analysis algorithm. However, tokenization in Arabic is not a straightforward step, according to [10]. This is complicated because of the structural sophistication of the Arabic language. Nevertheless, many tools are available for solving tokenization problems in Arabic, such as NLTK (Python) and PyArabic (Python). Finally, all reviewed resources tokenized the Arabic dataset-based words.

B. Stopword Removal

Stopword removal is essential for preprocessing all languages because it removes unimportant words and reduces the dimensionality of the texts used in the text analyzer. Stopwords include prepositions, pronouns, and articles that appear frequently and do not help distinguish documents [19]. Thus, the stopword benefits only from a syntactic procedure and does not distinguish the subject matter [20]. Furthermore, it is essential to filter out noise from vital text that is extraneous and irrelevant to the tasks [21]. Therefore, stop words are typically released with the support of a predefined list that includes common stop words.

Moreover, the most general strategy for stopword removal is based on frequency calculation and the removal of the most frequent words in all documents, which is known as dataset dependency [11], [20]. Another method to remove stop words is to use artificial patterns and perform entropy calculations. However, some general stop word lists can be used and are available online in many languages, including Arabic [20].

Stopword removal is applied to all languages as an essential step towards reducing the number of unimportant words. Many publications have discussed and provided solutions to this step in Arabic. For instance, [22] built a list containing 11,403 Arabic stop words for stop word removal. In contrast, the authors in [23] created an Arabic stopword list available online, consisting of three lists: a general list (1,377 stopwords), a corpus list that manually checked for the most repeated words appearing over 25,000 times (235 stopwords), and a combined list from the previous two. However, the authors of [18] removed stopwords based on the prepared list, and used the same stopword list as [23].

In contrast, the authors of [21] proposed a solution to the Arabic stopword construction problem by providing a comprehensive list called the Arabic stopword list (ASL) with a stopword analyzer. The analyzer connected the ASL list with machine learning to discover the most probable stop words, divided into three classes: native particles, special nouns, and special verbs for 3,931 stop words. The authors then derived a complex stop word list containing 67,153 words by adding all possible clitics to the simple stop word list. Finally, the authors tested the proposed list against the available list through a quantitative evaluation, revealing that the ASL exceeded other lists in terms of coverage. Compared with the proposed solution, the main stopword lists are Khoja [24], El-Khair [23], and the stopword project [2] and the Ranks NL[3]. Table I summarizes these stopword lists.

TABLE I. SUMMARY OF AVAILABLE STOPWORD LISTS

Stopword list name	**Description**	**Limitation**
Khoja	It contains 168 stopwords and was developed using statistical and rule-based techniques.	The list contains not all criticized forms of every stopword and corpus dependent.
El-Khair	It created three stopword lists: The first is a general stopwords list based on the Arabic language syntactic classes, consisting of 1,377 stopwords. The second is Corpus-based: manual checking of the words that appear more than 25,000, verifying this condition, and containing 235 words. Finally, the third list consists of 1,529 stopwords by combining the general list and the corpus-based lists	The list was not comprehensive and did not contain discretized.
Stopwords project	The Arabic list comprises 162 stopwords created under the GNU GPL v3 license.	The list corpus dependency contains some stopwords that may do not be correctly classified as stopwords, such as «««مليار، قوة، اعلنت»»» (billion, force, announced).
Ranks NL	The Ranks NL project was search engine optimization, but the project suggests an Arabic stopword list containing 102 stopwords.	Corpus was dependent.
ASL	The general list contained 3,931 stopwords. Then, the authors added all possible clitics to get the complex list, which contained 67,153.	

The authors of [12] used a general strategy to determine a stopword list by calculating the term frequency for all terms and then removing the most frequent words after sorting the terms. Nevertheless, the authors first prepared the Arabic text through the tokenization step and removed stop words; thus, they mainly examined and documented clustering. Similarly, the authors of [17] used a general strategy to remove stopwords and compared text preprocessing techniques in Arabic based on the frequency of words in the dataset. Consequently,

[2]http://code.google.com/p/stop-words
[3]http://www.ranks.nl/stopwords/arabic

they studied stopword removal and eliminated frequently used unwanted words.

Furthermore, the authors of [22] conducted scoping reviews over the past two decades (2000–2020) for Arabic topic identification, following the PRISMA-ScR guidelines. One stage that they focused on was the preprocessing step with feature extraction, which included stopword removal, stemming lemmatization, and feature selection. The authors provide an overview of the field to make it current. Finally, they recommended future work to enhance topic identification performance by working on preprocessing phases or implementing other algorithms.

Finally, stopword removal reduces text dimensionality because it removes words that frequently appear in the text without affecting text analysis, such as prepositions and pronouns. Moreover, stop-word elimination is recommended in most cases [22]. There are two strategies for stopword removal based on the reviewed papers: a general strategy based on word frequency or building, and using a general stopword list. Many general lists can be used and are available online, such as those published in 2006 by [25] containing 1,377 words. Moreover, a recent list was published in 2019 by [21] called the ASL list, consisting of three categories: native particles, particular nouns, and special verbs. Thus, the list contains 3,931 words.

C. Normalization

Text normalization is essential for text classification and analysis. Some normalization steps were applied to all languages, such as removing numbers and special characters. However, other normalization steps are language-dependent, because there is a specific way to normalize each language. For example, one step might include normalizing uppercase letters to lowercase letters in English, as there are no lowercase or uppercase letters in Arabic. Thus, normalization in Arabic related to normalizing letters includes normalizing different forms of alif (ا إ أ) to (ا) and removing diacritics that are not used in English [26].

Moreover, several normalization steps are typically executed to reduce the number of extracted terms. The significant normalization applied to Arabic is presented in the following steps, based on the reviewed papers (Fig. 2).

Thus, normalization differs from stopword removal. Normalization is related to formalizing the shape and form of words and other goals, whereas stopword removal focuses on removing unimportant words from the text. Because Arabic requires a unique normalizing process, many research publications have focused on normalization steps. Table II lists the most commonly used Arabic normalizations.

Normalization is a significant and essential step in most text analysis models. For example, the authors in [17] investigated the impact of 26 preprocesses applied to Arabic social media, particularly tweets, by training a classifier to identify health tweets. Fourteen of the 26 preprocesses focused on variants to normalize the Arabic letters. The authors studied a normalization step that converts a list of words into a more uniform sequence, such as removing punctuation, diacritics, repeating, and duplicating letters, including noise removal that seeks to destroy unwanted characters from the text, such as non-Arabic

TABLE II. THE MOST ARABIC NORMALIZATION THAT USED

#	Normalization step
1	Remove non-letters and special characters, such as $, &, and %
2	Remove non-Arabic letters
3	Replace initial (إ ،آ orأ) with bar alif (ا)
4	Replace Ta'a marbota (ة) with Ha'a (ه)
5	Remove the (ال) from the beginning of the word
6	Replace the final (ى) with (ا)
7	Remove the diacritics.

letters and numbers. They also described Arabic normalization steps related to Arabic letters, and the researchers normalized many letters, including two letters, five letters, and six letters, such as "Hamza," "alif" type, and "ta'a marbota."

Similarly, [10] provided an integrated solution for Arabic preprocessing in social media challenges by cleaning a dataset and normalizing Arabic social media text. The cleaning step focuses on cleaning noisy text by selecting only the required text instead of many processes to remove different data noise. The algorithm transforms each letter into its standard form during cleaning. For example, "alif" " أ" has several forms, which are " آ ،إ ، أ". In contrast, the normalization step changes the text into its standard form, with the algorithm changing a non-normal word into a normal word by eliminating duplicate characters and using a set of common non-normal words.

Furthermore, the authors of [18] normalized Arabic letters to help downgrade the various character shapes and produce a uniform shape representing these shapes. Therefore, the authors investigated the impact of stemming methods on feature reduction and classification accuracy.

Normalization is another essential step in preparing text for analysis. Some normalization processes are applied to all languages, whereas the other steps are language-dependent. Since Arabic has many characteristics related to letter shape, it requires normalization, including different forms of alif (ا إ أ) to (ا), Ta'a marbota (ة) to Ha'a (ه), and the final (ى) with (ا). Moreover, the normalization of Arabic involves removing the diacritics and the (ال) from the beginning of the word. Although several normalization steps are usually conducted to decrease the number of extracted terms, many available tools can be used for normalization problems, such as normalizing Arabic words using camel tools (Python) and PyArabic (Python).

D. Stemming

Stemming refers to the reduction of words to their stems or roots that are used to fit different word variants. Moreover, stemming is an essential preprocessing step for preparing text for the analyzer model, regardless of language type. There are three types of Arabic stemming, based on [27], [28], [6].

The first is root-based stemming, where the primary goal is to extract the roots of words using a stemmer. For example, in the Arabic language, several words can be reduced to one root, such as (التعليم ، العلماء،المعلم), which means "the education," "the scientists," and "the teacher," respectively, and are reduced to the root (علم), which means "science" [11], [28], [22].

Second, the stemmer eliminates additional suffixes and prefixes from words using light-stemming. Thus, it does not extract the original root; it only removes prefixes and suffixes from words. For example, the Arabic word (المعلم), which means "the teacher," is reduced to (معلم), which means "teacher." Hence, the semantics of the words are not affected by light stemming. Moreover, it normalizes words with diacritics and stretching characters (Tatweel ex: خبـــــــــــر converts to خبر) [11], [28], [22].

Finally, using a stem-based approach, the stemmer determines the stem that is part of the word to which grammatical prefixes and suffixes are added [28], [22]. The stem-based method does not attempt to determine the word root; it typically removes only clitics from a given word [28]. This technique preserves and represents the meaning of the text content well because it grammatically regroups inflected and related words [28], [22].

Stemming is a vital step in preparing text for analysis in most languages. Therefore, many studies have been conducted on Arabic stemming. For instance, a study [27] published in 2019 compared the stemmer ARLSTem with two versions (ARLSTem v1.0 and ARLSTem v1.1) to other light stemmers: the ISRI stemmer, Soori's stemmer, Assem's stemmer, and the Light10 stemmer. The authors then proposed an improved version of the ARLSTem stemmer by reordering some steps while adding others to improve performance. Next, the authors compared the improved version with the original and other stemmers, such as Light10. The results showed that the two versions of ARLSTem algorithm outperformed other algorithms based on understemming errors and overstemming indices.

Similarly, the authors in [10] provided an integrated solution for Arabic preprocessing of social media challenges by applying their new stemming algorithm to social media containing standard Arabic and dialects. The new Arabic stemmer module generates word stems, word roots, and specific word identifiers. The proposed model solved Arabic social media challenges that help understand a word's meaning by providing its root and determining whether the word is a noun, stop stopword, non-standard Arabic word, dialect, error, or non-Arabic word.

In comparison, [18] investigated the impact of stemming methods, an essential preprocessing step in the classification accuracy, and the number of features used. The primary preprocesses used were tokenization, normalization, stopword removal, feature extraction, and stemming. First, the authors used the Light10 stemmer for stemming and represented documents as vectors. They then conducted experiments to test the impact of the stemming algorithms on the K-nearest neighbor, naive Bayes, and decision tree classification algorithms. The results showed the effectiveness of the stemming algorithm in reducing half of the features used while improving accuracy.

Similarly, in [28], the authors compared Arabic topic identification to investigate different stemming algorithms. The main objective was to study the effects of stemmers: root-based, light stem, and stem-based. The authors used primary and available steaming approaches for the root-based selection of the Khoja and Tashaphyne root approaches. In contrast, light stem used Light10 and Tashaphyne light approaches, whereas for other types, they selected Farasa and Alkhalil1. The authors experimented using latent Dirichlet allocation (LDA) to identify the topics with different Arabic stemming algorithms. Hence, the authors determined which forms of words—root, stem, and light—were affected by the preprocessing step in topic identification. Indeed, Light10 identified topics better than the other stemmers.

Furthermore, [12] examined and compared text-preprocessing techniques in Arabic document clustering. The study indicated the significant impact of term pruning with stemming and term weighting as preprocessing steps in studying text preprocessing effectiveness. Notably, the authors experimented with different preprocessing techniques with term pruning combinations using the three systems. The first system combined term pruning with term weighting and light stemming (Light10). The second system combines term pruning with term weighting and normalization, whereas the last system combines term pruning with term weighting, normalization, and Khoja root-based stemming. The results were based on precision, recall, and F-measures, as the evaluation measures showed that the system using light stemming achieved better results than the others.

Similarly, [17] investigated the impact of the 26 preprocesses applied to Arabic health tweets by training four classifiers: MNB, logistic regression, linear SVC, and KNN. They discussed the steaming process and classified steaming into three types—root, light, and lemmatization—and found that stemming techniques increased the accuracy of the classifier.

Additionally, [22] investigated the effectiveness of Arabic preprocesses for text classification to demonstrate that properly selecting preprocessing techniques leads to a positive Arabic classification outcome. The preprocess techniques mentioned and discussed are stopword removal, stemming, and lemmatization. Moreover, they studied the effects of different combinations of these techniques to determine the best performance using ARLSTem v1.08. It was the best stemmer based on the surveyed papers that demonstrated positive outcomes in Arabic text classification, whereas the lemmatizer used MADAMIRA v2.1. The authors concluded that the best performance occurred when applying the three preprocess techniques were applied: stopword, stemming, and lemmatization.

Finally, the authors in [29] conducted scoping reviews over the past two decades (2000–2020) for Arabic topic identification following PRISMA-ScR guidelines, focusing on the preprocessing step. The preprocessing step and feature extraction included stopword removal, stemming lemmatization, and feature selection. Based on the authors' reviewed papers, the best algorithms were recommended for the steaming process. Common stemmers successfully used in Arabic topic identification are ARLSTem, Tashaphyne light stemmer, Farasa, Khoja stemmer, Light10, Al Khalil Morph Sys, Assem's stemmer, Soori's stemmer, and the ISRI stemmer. Finally, the authors suggest future research to enhance the topic identification performance by working on the preprocessing phases and implementing other algorithms.

Thus, stemming algorithms aim to remove suffixes, infixes, and other letters to grammatically reduce the multiforms of the same word in texts while reducing features. Moreover, stemming is an essential preprocessing step, regardless of the language type. Table III summarizes the main stemming algorithms used in Arabic.

TABLE III. SUMMARY OF AVAILABLE STEMMING TOOLS

Stemming name	Stemming type	Resource
Khoja	Root-based	[30]
ISRI	Root-based	[31]
Light10 (there is the previous version of Light 1 until 9)	Light stemming	[32]
ARLSTem	Light stemming	[33], [27]
Assem's stemmer	Light stemming	[34]
Tashaphyne	Light stemming	[35]
Farasa	Stem-based	[36]

Based on the reviewed papers, two Arabic stemming types are widely used: light stemming and root-based stemming. According to [28]. Moreover, it is more efficient than topic identification. Hence, the best stemming algorithm for clustering is light stemming, whereas the two best available stemming algorithms are ARLSTem and Light10 stemming, based on reviewed papers, such as [28], [18].

Additionally, based on reviewed papers, root-based stemming has been used by some authors. The Khoja stemmer, which removes the longest suffix and prefix, is a well-known root-based stemmer. Subsequently, the root dictionary matches the remainder with verbal and noun patterns [28]. At the same time, the Tashaphyne root stemmer is another famous root-based stemmer for Arabic. This stemmer identifies the root for removing prefixes and suffixes based on a default or two customized lists of prefixes and suffixes [28]. Moreover, the Tashaphyne stemmer identifies the light stems of words.

E. Vectors

Feature extraction (vectorization) is an essential preprocessing step, and the last step is performed before the analyzer algorithm. The feature extraction process transforms the text into vectors [26]. Moreover, the two most commonly used methods for extracting features from text are term frequency-inverse document frequency (TF-IDF) and Bag of Words (BoW) [26], [37]. These methods are among the most popular for computing term weights [22], [28]. Table IV summarizes the main differences between the extraction methods.

TABLE IV. SUMMARY OF VECTOR TECHNIQUES

	BoW	TF-IDF
Focus	It constructs a collection of vectors, including the word count of occurrences in the document.	TF-IDF includes information on the more and less important words ones.
Main information	BoW vectors are straightforward to interpret.	Usually acts better in the machine learning approach.
Drawbacks	Most avoid BoW and TF-IDF techniques when understanding the context of words most involved.	
Widely used for text analysis	Used but less than TF-IDF	Yes

The TF-IDF method comprises the term frequency (TF) and inverse document frequency (IDF). The TF is related to calculating the token frequency occurrence in a document, which propagates proportionally with the document size. Therefore, tokens occur more frequently in long documents than in shorter ones. The second part, the IDF, focuses on weighing down frequent tokens while scaling up rare tokens by calculating the importance of a word in all documents [17], [29], [37].

Second, the BoW is the most straightforward representation of text in terms of numbers. This method represents a sentence as a BoW vector; therefore, the sentence is presented as words with a number to indicate the occurrence in a sentence, such as TF. For example, when applying a BoW to documents (text), the result is a matrix based on all terms as columns and appears in each document as a row. Therefore, the sparse matrix result contains many 0s because it combines all the vectors [22], [29], [37].

The main drawback of using the BoW model is that the vocabulary size increases if new sentences contain new words. Hence, the lengths of the vectors also increase. Finally, BoW does not retain information related to the grammar of a sentence or the ordering of words in the text [22], [29].

Vectorization is vital for preparing text for text analysis in most languages. Many studies on Arabic feature extraction (vectors) related to this step have been published. For example, the authors in [12] investigated and compared text preprocessing techniques in Arabic document clustering and studied the effectiveness of the following text preprocessing techniques: term pruning, steaming, and term weighting based on TF-IDF. The authors tested the preprocessing combinations using these three systems. The first system included term pruning with term weighting (TF-IDF) and light stemming. The second combines term pruning with term weighting (TF-IDF) and normalization. The last system combines term pruning with term weighting (TF-IDF), normalization, and root-based stemming. Thus, this study demonstrates the significant impact of term pruning with light stemming and TF-IDF. Furthermore, the authors in [17] investigated the impact of 26 preprocesses applied to the Arabic healthcare tweet classifier and mentioned that the most extracted feature methods used were BoW and TF-IDF. However, they applied TF-IDF as a feature-extraction preprocess to transform the text into vectors.

The authors of [22] explored the significance of Arabic preprocessing for text classification to prove that proper selection of preprocessing techniques leads to a positive Arabic classification result. Moreover, the authors investigated the consequences of different combinations of preprocessing techniques to determine the best performance. They applied feature extraction TF-IDF and, as feature selection, used chi-square, then ran the classifiers. Additionally, the authors in [18] investigated the impact of stemming methods on classification accuracy and the number of features used. Feature extraction was one of the main preprocesses used in this study. The authors represented the documents as vectors using TF-IDF.

Finally, [29] followed the PRISMA-ScR guidelines to conduct scoping reviews over the past two decades (2000–2020) of Arabic topic identification. The authors focused on the preprocessing step, and they mentioned that the most standard techniques used for feature extraction included BoW, Bag of Concepts (BoC), count vectorizer, TF-IDF vectorizer, and chi-square test ($\tilde{\chi}^2 test$). Finally, the authors recommend future research to enhance topic identification performance by working on the preprocessing phases and implementing other algorithms.

Thus, vectorization is vital for converting textual features into numerical vectors. Vectorization is the final step in text-analysis models, and classification and clustering are essential. The TF-IDF method is the most widely used method for Arabic clustering and document analyses. Based on surveys and literature review, this method has proven to be the most effective.

F. Feature Selection

Feature selection methods have been verified as valuable for text classification and clustering. There were three feature types: irrelevant, powerfully relevant, and weakly relevant. Moreover, the clustering approach requires a suitable feature selection method that reduces the selection of irrelevant features to represent the text data [38]. The feature selection method removes irrelevant and duplicate features from datasets and retains features that include reliable and helpful information to reduce the dimensionality of the text in clustering techniques [28], [38]. Consequently, high-dimensional data affect the efficiency and performance of clustering approaches, dramatically decreasing the efficiency and increasing the execution time [38]. Feature selection algorithms can be classified into two primary categories: filtering and wrapping [9], [38].

The most well-known methods used for feature selection are chi-square and information gain (IG). The chi-square test is a filtering method for selecting features. Chi-square is a straightforward and computationally fast method. It can deal with a sizable dimensional feature and has proven its efficiency mainly when applied with the TF-IDF extracted feature technique [29], [38].

Similar to the work of [38], the researchers extracted features from the training datasets by calculating the TF-IDF score for each feature and then applied the chi-square test to select relevant features and eliminate irrelevant features. The chi-squared test was efficient. The chi-square test tests the independence between the occurrence of a specific feature and the occurrence of a specific type. The null hypothesis of the chi-square test was that no relationship existed between the two variables. Therefore, they are independent of each other.

The authors of the review paper [29] discussed the chi-square test as a feature selection method used for Arabic topic identification. The authors conducted scoping reviews for Arabic topic identification following the PRISMA-ScR guidelines over the past two decades (2000–2020), focusing on four phases of feature selection.

IG was used to discover the most relevant features in the label dataset. The IG ranks these features based on their entropy values, which reflect the impact of a unique feature on deciding the type label. The most relevant features are selected based on a predefined threshold value. As in [26], the authors first built the TF-IDF matrix by applying the feature extraction method and then used the IG method as a filtration step to select the most relevant Arabic features based on their ranks. This filtering step is essential for reducing the spatial dimensionality of the classifier by eliminating all features with IG ranks below a given threshold value. In [39], the authors used an IG feature ranking technique to remove irrelevant features. Furthermore, IG was used to reduce the size of the features and select the top-ranked features to train the classifier.

Term pruning is a feature selection that provides a helpful step based on a survey that eliminates words with counts less or greater than a typical threshold. For example, an experimental survey [18] showed that the best practical pruning factor is a minimum of three. Thus, pruning aims to remove any word that appears fewer than three times and is considered unimportant, thereby reducing its features.

The authors of [12] examined and compared text preprocessing techniques in Arabic document clustering and demonstrated the significant impact of term pruning on different preprocessing techniques. They experimented with different combinations of preprocessing techniques with term pruning using three, five, seven, and nine terms in the three systems. The first system combined term pruning with term weighting and light stemming (Light10), whereas the second system combined term pruning with term weighting and normalization. The last system combined term pruning with term weighting, normalization, and Khoja root-based stemming. The results were based on the precision, recall, and F-measure as evaluation measures. They showed that term pruning with a minimum of three combinations of term weighting-based TF-IDF and light stemming achieved better results than the others.

G. Lemmatization

Text lemmatization seeks to regroup semantically associated words. Unfortunately, lemmatization is limited as a preprocessing task in Arabic text classification, and no study has used lemmatization for clustering based on reviewed and surveyed papers because it is a complicated level of text processing. Moreover, most Arabic lemmatizers are royal and not publicly available, unlike Arabic stemmers. Nevertheless, some studies have reported lemmatization efficiency, particularly for information retrieval, text summarization systems, and text indexation [22].

Typically, document classification is straightforward when the meaning of the content is well represented. Thus, lemmatization regroups equivalent written words semantically in various syntactic structures and relates them to their ecclesiastical base representation called a "lemma" (i.e., a dictionary reference form). Thus, applying lemmatization to text classification as a preprocessing task is particularly beneficial [22].

The authors of [29] used text lemmatization plans to regroup semantically associated words written in various syntactic forms and associate them with their lemma. Unlike stemming, lemmatization is reasonably limited to Arabic preprocess because most Arabic lemmatizers are royal and not publicly available. However, Farasa[4] and MADAMIRA[5] are tools available for lemmatizers.

[4]https://alt.qcri.org/farasa/

[5]http://innovation.columbia.edu/technologies/cu14012_arabic-language-disambiguation-for-natural-language-processing-applications?license=108

Thus, lemmatization preprocessing steps may be helpful, based on survey papers, similar to [22], [17]. For instance, the authors of [22] investigated the effectiveness of Arabic preprocesses for text classification and found that appropriate preprocessing techniques lead to an optimistic Arabic classification outcome. Hence, they studied the effects of different combinations of the following techniques to determine the best performance: stopword removal, stemming, and lemmatization. The authors used MADAMIRA v2.1 as a lemmatizer because it was the best lemmatizer based on surveyed papers that demonstrated positive outcomes in Arabic text classification. The authors identified the best performance among the three preprocess techniques together.

Similarly, [17] investigated the impact of the 26 preprocesses applied to Arabic healthcare tweets by training four classifiers: MNB, logistic regression, linear SVC, and KNN. The authors discussed the effect of preprocessing on the classifier, one of which was lemmatization. They found that the lemmatization stemming type performed well for all four classifier models.

Furthermore, the authors in [29] conducted scoping reviews for the past two decades for Arabic topic identification based on the PRISMA-ScR guidelines to provide new researchers in the field and advanced practitioners with a closer look at improvements in Arabic topic identification in the last decade, while suggesting several recommendations for better Arabic topic identification. The authors focused on several stages, one of which was preprocessing. The preprocessing step and feature extraction mentioned in [29] involve stopword removal, stemming lemmatization, and feature selection. Based on the authors and reviewed papers, a few Arabic lemmatizers are available for free. The best algorithms recommended for lemmatization by the authors are Farasa and MADAMIRA. Finally, the authors suggest future research to improve topic identification performance by performing preprocessing phases or implementing other algorithms.

Finally, some preprocessing steps may be helpful based on the survey papers, but they are not essential for preparing the Arabic text. One of these steps is lemmatization. The lemmatization step aims to regroup semantically associated words written in different syntactic forms, associate them with their lemmas in the text, and reduce their features. Unlike stopword removal and stemming, the use of lemmatization for Arabic preprocessing is reasonably limited because most Arabic lemmatizers are proprietary and not publicly available [29]. However, the lemmatization process improves performance as a preprocess but is rarely used by researchers because research in this area has not been published or is freely available online. The only two tools available in Arabic are Farasa and MADAMIRA. Thus, this area requires further research and proposed solutions public to the research community.

H. Synonyms

One Arabic language characteristic is morphological richness, wherein the same verb can have thousands (literally) of different forms [40]. Table V summarizes these methods, and the following paragraphs summarize the most widely available methods and tools for finding and creating synonyms for Arabic.

TABLE V. SUMMARY OF SYNONYM TECHNIQUES

	Sources	**Techniques**	**Resource**
WordNet [41]	The AWN was created based on Princeton WordNet (PWN) strategy and contents.	Translation and manually checked.	Open
word2Vec [42]	Twitter and Wikipedia.	CBoW and Skip-Gram have different n-gram and unigram features.	Open
FastText [43]	Common Crawl and Wikipedia.	CBoW with sub-wording techniques.	Open

The first method to find and create synonyms for Arabic is Word2Vec, an efficient solution to synonym problems that leverages the context of the target words proposed by Google. There are two types of Word2Vec: skip-gram and a Continuous Bag of Words (CBoW) [40], [41], [44]. The following paragraphs briefly describe these two methods.

In skip-grams, the input is the center word (target), whereas the outputs are the words surrounding the target words. For example, in the sentence “I have a pretty cat,” assuming the window size is 5, the input for the neural network would be “a” whereas the output would be “I,” “have,” “pretty,” and “cat.” The network contained one hidden layer with dimensions equal to the embedding size, which was smaller than the input/output vector dimension. The output layer is a softmax activation function so that each part of the output vector represents how a specific word will probably occur in the context. Word embedding for the center words can be obtained by extracting the hidden layers after providing a one-hot expression of that word into the network [40], [42], [45].

Furthermore, the vectors are more “significant” in describing word relationships. For example, vectors obtained by removing two related words sometimes represent meaningful concepts such as gender or verb tense. Finally, for all input and output data, most studies used exact dimensions and one-hot encoding [40], [42], [45].

While CBoW is similar to skip-gram, the inputs are the words surrounding the center words, whereas the output is the center word (target). The idea is that given a context, they like to know which word is most likely to occur. Thus, it aims to discover embeddings by indicating the center word in a context that provides other words in the context without respect to their order in the sentence [40], [42], [45].

The most significant distinction between skip grams and CBoW is the manner in which word vectors are generated. The CBoW model involves all samples with the target word in the neural network, and then takes the average of the extracted hidden layer. For example, assume only two sentences in a dataset: “He is a friendly man” and “She is a smart princess.” To compute the word representation for the word “a,” two examples are needed: “He is a friendly man” and “She is a smart princess.” These are placed in a neural network that takes the average value of the hidden layer [40], [42], [45].

The second method for determining synonyms is FastText, an extension of Word2Vec, as suggested by Facebook in 2016. FastText breaks words into several n-grams (subwords) instead of providing one word to the neural network. Infrequent words are then adequately represented, because it is highly likely that some of their n-grams will also appear. For instance, the tri-grams for the word fruit are “fru,” “rui,” and “uit.” The word-embedding vector for the fruit was the sum of these n-grams. After training the neural network, the results are word embeddings for all n-grams in the training dataset [40], [46], [43].

Finally, WordNet is used for constructing lexical resources for SA. The Arabic WordNet (AWN) is based on the universally accepted Princeton WordNet (PWN) strategy and content. It enables translation into English and dozens of other languages at the lexical level. In addition, it encodes language-specific concepts and links as required or preferred. The results are called core word nets for Arabic with the essential system embedded in a solid semantic framework [41].

Moreover, the enrichment of AWN was published by [47], and the latest version was constructed by semi-automatically expanding its content, adapting and using existing approaches and resources generated for other languages to expand AWN [32]. Thus, AWN is a lexical dictionary or database utilized to discover synonyms and determine different relations among Arabic words containing several elements, including nouns, verbs, adjectives, and adverbs, which vary into sets of cognitive concepts (i.e., synsets) [48].

Because Arabic is rich in synonyms, a possible way to improve clustering performance is to reduce synonym words to a single word. Thus, one of the main goals of this survey is to determine whether anyone has used the synonym method as a preprocessing step to reduce the number of synonyms to help reduce the features. Based on the reviewed papers, this study found that none of these methods were used as preprocessing steps. Nevertheless, synonym methods and techniques have been used to build lexical or corpora [49], which are used to build medical datasets, or to build a corpus for some approaches such as sentiment analysis, similar to the authors in [17]. This idea may need more research in the future to satisfy and clarify whether synonym methods can be used to reduce the features by replacing the synonym with one word, thereby demonstrating that this technique reduces the feature’s dimensionality and improves clustering quality.

I. Semantics

In this study, Arabic semantics are related to Arabic semantic ambiguity, mainly Arabic word sense disambiguation (WSD), a task that seeks to determine the meaning of a word given its context [50]. Arabic semantic obscurity depends on the meaning of a word or many words that can be misconstrued. Obscurity can come from the order of words or word types as verbs or nouns [51]. Moreover, Arabic, which uses diacritics, such as the Arabic read word (علم), can take many meanings based on how diacritic (عُلِمْ: understood), (عَلَم: flag), (عَلَمْ: teach), and (عِلْم: science).

Semantics and meaning can be classified as branches of linguistics. Semantics and meaning are associated; some use semantics to serve meaning, whereas others use meaning to serve semantics. The first states that science investigates the requirements provided by linguistic symbols to carry out meaning. In contrast, the second emphasizes the importance of overt semantics in hidden meanings. However, another definition of semantics is the branch of linguistics that attempts to investigate changes in meaning via the analysis of linguistic structure phonetically, morphologically, lexically, and syntactically, while considering changes in usage over time [51].

Many studies have investigated this issue. For example, [50] published a 2019 review paper discussing Arabic word meaning disambiguation to inspire readers to solve Arabic words with morphological and semantic ambiguities. The morphological challenge of the Arabic language is still lacking since more than ten variations can be assigned to a non-vocalized Arabic word, for example, the words “““شَعْر””” (hair) or “““شِعْر””” (poetry). Moreover, the authors investigated numerous studies in this field by analyzing their assessment methods and linguistic resources (e.g., corpora and lexicons), recommending solutions to current semantic problems, and suggesting future directions to improve research in this area.

Thus, the Arabic language has a problem related to semantic obscurity because semantic ambiguity is associated with the meaning of a word, which can be misunderstood. Obscurity can be reached from the order of words or word types as verbs or nouns [42]. Therefore, one way to improve the clustering quality and performance is to use semantic solutions as a helpful method to solve ambiguous words when the word has the same form but different meanings based on sentences and semantics. Similar to [52], the CBOW model was used to capture the semantic relationship between the terms (word tokens), followed by using K-means to identify essential documents for Arabic summarization.

This is similar to the results of [53], who proposed a solution for Arabic Word Sense Induction (WSI) in NLP, mainly when applied to sentiment analysis. It presents a two-stage approach; the first stage uses a Transformer-based encoder like BERT or DistilBERT to encode the input sentence into context representations. In the second stage, k-means clustering and agglomerative hierarchical clustering (HAC) were applied to the embedded corpus obtained in the first stage. The proposed approach improves existing methods and revolutionizes WSI research.

Hence, one of the main goals of this survey was to determine whether anyone had used the semantic method as a preprocessing step to reduce word ambiguity and improve clustering quality. However, based on the reviewed papers, no study has used this feature as a preprocessing step. Moreover, semantic methods have been used in separate research fields using a domain-based approach. Semantic ambiguity in Arabic has recently become an active research topic. However, perhaps using the semantic ambiguity solution to improve Arabic clustering quality requires more research to satisfy and clarify whether a semantic method and word ambiguity list can improve clustering quality.

IV. Recent Arabic Clustering Publication

This section reviews the recent Arabic clustering research papers and articles published between 2019 and 2024. It summarizes the preprocessing steps used by the authors to prepare the Arabic text, as seen in Table VI. The following section discusses the main points extracted from Table VI below.

All reviewed papers used the stemming and converting the results into TF-IDF vectors. Most of the reviewed documents (91%), used the word tokenization step and removal of stopwords, while (66%) used Arabic normalization. All reviewed papers used TF-IDF as a feature representation for clustering Arabic documents. Only two reviewed papers used bigram to combine two terms as a collocation technique to better understand semantic and meaning problem—only one reviewed paper used term pruning to eliminate low-frequency words.

Regarding the feature selection method, only two studies used IG; however, they focused on the Arabic classification-based bio-inspired method. In addition, based on the reviewed papers, only one study used dimension-reduction methods as a preparation step for Arabic clustering. None of the reviewed documents applied semantic methods for Arabic clustering, or even importing, to overcome contextual meaning. None of the reviewed studies applied synonymous methods to Arabic clustering.

Fig. 3 summarizes the percentages of used main preprocess steps related for the Arabic normalization, stemming, and feature selection extracted from Table VI. Fig. 3(A) shows that more than half of the reviewed Arabic research papers 67% applied normalization preprocess for Arabic text before applying a text analyzer. In addition, Fig. 3(B) shows that all of the reviewed Arabic research papers applied the stemming method to prepare Arabic texts. Most stemming methods used root stemming (54%), whereas 46% used light stemming. Also, one research paper applied both types of stemming. Finally, Fig. 3(C) shows that 17% of the review papers applied feature selection methods exactly applied IG. In contrast, the remaining 83% did not apply feature selection methods to prepare the Arabic texts.

V. Discussion

This survey intensively investigated and focused on many Arabic preprocessing steps as references for researchers engaged in Arabic preprocessing for clustering. The survey was concluded by answering the questions raised at the beginning of this paper based on the results obtained. What preprocessing steps are used for the Arabic language in general and the cluster, especially? Preprocessing researchers have used tokenization, normalization, stopword removal, stemming, and vectorization. Some researchers have used feature selection to prepare text for a cluster to reduce the number of dimensions. In contrast, lemmatization, synonyms, and semantics were not used as preprocessing for clustering.

The second question, What tools are available for the algorithms and techniques for each preprocessing step? Many resources have been proposed for each preprocessing step; however, some preprocesses, such as lemmatization, require more investment and improvement. In addition, as a future recommendation, you can propose a stopword removal list for many fields because each domain has its own list.

In addition, the last two questions, Does preprocessing address the richness of synonyms in Arabic and can it be used to reduce dimensions? Does preprocessing address semantics as a primary characteristic of Arabic? This survey found that no one had used the synonym method as a preprocessing step to reduce features. In addition, no one has used the semantic method as a preprocessing step to reduce word ambiguity and improve clustering quality. Additionally, this survey suggests conducting further research to address the richness of synonyms in Arabic to reduce the dimensions by combining many synonyms with the same meaning. In addition, the researcher may deal with semantics to improve clustering quality as a preprocessing process to improve step. Therefore, future research should investigate the effectiveness of using synonymous techniques and semantic solutions to reduce the dimensions of complexity or improve the quality of cluster results.

Therefore, this study concludes that preprocessing steps are preferable because they provide benefits and improve the quality of an analysis model based on the surveyed papers. Based on the reviewed research papers, Arabic preprocessing is an active research area. Nevertheless, this field requires further research, especially clustering, to provide more solutions and publicly available tools to benefit the Arabic research community.

VI. Conclusion

Arabic is a complex language because of its unique orthography, grammar, and punctuation. A sentence contains a verb, a subject, or an object. Words are derived from trilateral or quadrilateral systems, making them challenging for algorithms and computational systems to understand. Moreover, Arabic has extensive synonyms and the meanings of Arabic words depend on the semantics of the sentences. For all these reasons, the analysis algorithms require specific Arabic preprocessing techniques.

Arabic text preprocessing methods are used in clustering to reduce the size of documents, simplify feature selection, and enhance processing speed. Hence, the primary purpose of a text preprocessing task in clustering is to address the considerable dimensionality problems. Therefore, several preprocessing techniques have been proposed.

This survey intensely investigated and focused on many Arabic preprocessing steps as a reference for researchers engaging in Arabic preprocessing for clustering. These preprocesses include tokenization, normalization, stopword removal, stemming, vectorization, lemmatization, feature selection, synonyms, and semantics. Moreover, this survey classified these preprocesses as essential or preferable steps.

Therefore, this paper concludes that preprocessing steps are preferable since they may benefit some models and improve the quality based on the surveyed papers. Additionally, this survey suggests conducting further research to deal with a richness of synonyms for Arabic to reduce the dimension and deal with semantics to improve the clustering quality as a preprocessing process. Indeed, based on the surveyed papers, no one to date has dealt with these characteristics to improve clustering quality.

The Arabic preprocessing step is an active research area based on the reviewed research papers. Nevertheless, this field needs more research, especially for clustering, to provide more solutions and publicly available tools to benefit the Arabic research community.

TABLE VI. Summary of Arabic clustering research papers.

Ref	Year	Tokenization	Stopword removal	Normalization	Stemming	Feature selection	Synonyms	Collection: Bigram, trigram	Vectors	Final output	Addition step
[54]	2019	Yes (word)	Yes	No	ISRI (Root)	No	No	Sentence level	TF-IDF	vectors	Sentence segmentation.
[9]	2019	Yes (word)	Yes	No	Yes (Light Stemming)	No	No	No	TF-IDF	VSM	Segmentation
[39]	2019	Yes (word)	Yes	Yes	Yes (Root)	Yes (IG)	No	No	TF-IDF	Matrix vectors	
[55]	2020	Yes (word)	Yes	Yes	Yes (Root)	No	No	Yes (Bigram)	TF-IDF	VSM	
[26]	2020	Yes (word)	Yes	Yes	Yes (Root)	Yes (IG)	No	Yes (Bigram)	TF-IDF	VSM	
[56]	2020	Yes (word, phrase)	Yes	Yes	Yes (ISRI)	No	No	No	TF-IDF	Matrix vectors.	Dimensional reduction.
[57]	2020	Yes (word)	Yes	No	stemming (Light10)	No	No	No	TF-IDF	document-term matrix	
[58]	2021	No	Yes	Yes	Yes (Light)	No	No	No	TF-IDF	VSM	Eliminate insignificant words
[59]	2022	Yes (word)	Yes	No	Yes (root and light)	No	No	No	TF-IDF	VSM	
[60]	2023	Yes (word)	Yes	Yes	Yes (Light Stemming)	No	No	No	TF-IDF	vectors	Arabic-BERT model
[61]	2023	Yes(word)	NO	Yes	Yes (ISRI)	No	No	No	TF-IDF and CountVectorizer	vectors	
[62]	2024	Yes	Yes	Yes	Yes (light stemmer)	No	No	No	TF_IDF	vectors	Aravec pre-trained word embedding.

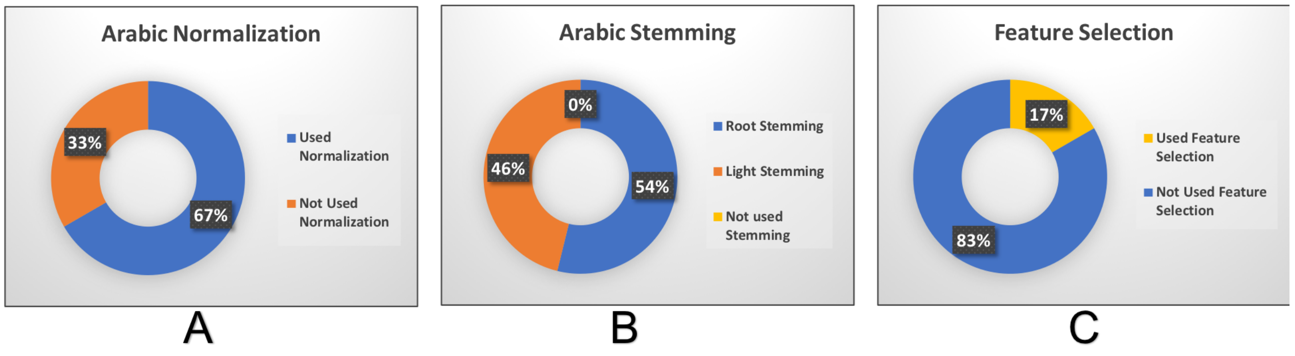

Fig. 3. (A) Percentage of reviewed Arabic papers that used the normalization step to prepare Arabic text. (B) Percentage of reviewed Arabic papers that used the stemming method with different types to prepare Arabic text. (C) Percentage of reviewed Arabic papers that used the feature selection step to prepare Arabic text.

References

[1] W. Hadi, Q. A. Al-Radaideh, and S. Alhawari, “Integrating associative rule-based classification with Naïve Bayes for text classification,” *Applied Soft Computing*, vol. 69, pp. 344–356, 2018.

[2] S. Bahassine, A. Madani, and M. Kissi, “Arabic text classification using new stemmer for feature selection and decision trees,” *Journal of Engineering Science and Technology*, vol. 12, pp. 1475–1487, June 2017.

[3] S. L. Marie-Sainte, N. Alalyani, S. Alotaibi, S. Ghouzali, and I. Abunadi, “Arabic Natural Language Processing and Machine Learning-Based Systems,” *IEEE Access*, vol. 7, pp. 7011–7020, 2019.

[4] S. L. Marie-Sainte and N. Alalyani, “Firefly Algorithm based Feature Selection for Arabic Text Classification,” *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 3, pp. 320–328, March 2020.

[5] I. Guellil, H. Saâdane, F. Azouaou, B. Gueni, and D. Nouvel, “Arabic natural language processing: An overview,” *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 5, pp. 497–507, June 2021.

[6] A. Alothman and A. Alsalman, “Arabic morphological analysis techniques,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, 2020. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2020.0110229

[7] A. Hotho, A. Nürnberger, and G. Paass, “A Brief Survey of Text Mining,” *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*, vol. 20, pp. 19–62, 01 2005.

[8] M. Alhawarat and M. Hegazi, “Revisiting K-Means and Topic Modeling, a Comparison Study to Cluster Arabic Documents,” *IEEE Access*, vol. 6, pp. 42 740–42 749, 2018.

[9] A. K. Sangaiah, A. E. Fakhry, M. Abdel-Basset, and I. El-henawy, “Arabic text clustering using improved clustering algorithms with dimensionality reduction,” *Cluster Computing*, vol. 22, no. 2, pp. 4535–4549, March 2019.

[10] M. O. Hegazi, Y. Al-Dossari, A. Al-Yahy, A. Al-Sumari, and A. Hilal, “Preprocessing Arabic text on social media,” *Heliyon*, vol. 7, no. 2, p. e06191, February 2021.

[11] A. S. Daoud, A. Sallam, and M. E. Wheed, “Improving Arabic document clustering using K-means algorithm and Particle Swarm Optimization,” in *2017 {Intelligent} {Systems} {Conference} ({IntelliSys})*, September 2017, pp. 879–885.

[12] M. A. Alhanjouri, “Pre Processing Techniques for Arabic Documents Clustering,” *International Journal of Engineering and Management Research (IJEMR)*, vol. Volume: 7, Number: 2, no. Volume: 7, Number: 2, 2017.

[13] K. Darwish, N. Habash, M. Abbas, H. Al-Khalifa, H. T. Al-Natsheh, H. Bouamor, K. Bouzoubaa, V. Cavalli-Sforza, S. R. El-Beltagy, W. El-Hajj, M. Jarrar, and H. Mubarak, “A panoramic survey of natural language processing in the Arab world,” *Commun. ACM*, vol. 64, no. 4, pp. 72–81, mar 2021.

[14] A. A. Al-Khulaidi and S. M. Yaseen, “Comparative Analysis and Evaluation of Stemming and Preprocessing Techniques for Arabic Text,” *Sana’a University Journal of Applied Sciences and Technology*, vol. 1, no. 4, 2023.

[15] Z. Alyafeai, M. S. Al-shaibani, M. Ghaleb, and I. Ahmad, “Evaluating various tokenizers for Arabic text classification,” *Neural Processing Letters*, vol. 55, no. 3, pp. 2911–2933, 2023.

[16] M. A. Attia, “Arabic tokenization system,” in *Proceedings of the 2007 {Workshop} on {Computational} {Approaches} to {Semitic} {Languages} {Common} {Issues} and {Resources} - {Semitic} ’07*. Prague, Czech Republic: Association for Computational Linguistics, 2007, p. 65.

[17] Y. Albalawi, J. Buckley, and N. S. Nikolov, “Investigating the impact of pre-processing techniques and pre-trained word embeddings in detecting Arabic health information on social media,” *Journal of Big Data*, vol. 8, no. 1, p. 95, December 2021.

[18] J. Atwan, M. Wedyan, Q. Bsoul, A. Hammadeen, and R. Alturki, “The Use of Stemming in the Arabic Text and Its Impact on the Accuracy of Classification,” *Scientific Programming*, vol. 2021, pp. 1–9, November 2021.

[19] H. Ahonen, O. Heinonen, M. Klemettinen, and I. Verkamo, “Applying Data Mining Techniques in Text Analysis,” Citeseer, Tech. Rep., 1997.

[20] A. Alajmi, E. Saad, and R. R. Darwish, “Toward an ARABIC Stop-Words List Generation,” *International Journal of Computer Applications*, vol. 46, pp. 8–13, January 2012.

[21] D. Namly, K. Bouzoubaa, R. Tajmout, and A. Laadimi, “On Arabic Stop-Words: A Comprehensive List and a Dedicated Morphological Analyzer,” in *Arabic {Language} {Processing}: {From} {Theory} to {Practice}*, K. Smaïli, Ed. Cham: Springer International Publishing, 2019, pp. 149–163.

[22] E. kah Anoual and I. Zeroual, “The effects of Pre-Processing Techniques on Arabic Text Classification,” *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, pp. 41–48, February 2021.

[23] I. A. El-Khair, “Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study,” February 2017.

[24] S. Khoja, “APT: Arabic part-of-speech tagger,” in *Proceedings of the Student Workshop at NAACL*, 2001, pp. 20–25.

[25] I. A. El-Khair, “Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study,” *International Journal of Computing &amp; Information Sciences*, vol. 4, pp. 119–133, January 2006.

[26] A. Alzaqebah, B. Smadi, and B. H. Hammo, “Arabic Sentiment Analysis Based on Salp Swarm Algorithm with S-shaped Transfer Functions,” in *2020 11th {International} {Conference} on {Information} and {Communication} {Systems} ({ICICS})*, April 2020, pp. 179–184.

[27] K. Abainia and H. Rebbani, “Comparing the Effectiveness of the Improved ARLSTem Algorithm with Existing Arabic Light Stemmers,” in *2019 {International} {Conference} on {Theoretical} and {Applicative} {Aspects} of {Computer} {Science} ({ICTAACS})*, vol. 1, December 2019, pp. 1–8.

[28] M. Naili, A. H. Chaibi, and H. H. B. Ghezala, “Comparative Study of Arabic Stemming Algorithms for Topic Identification,” *Procedia Computer Science*, vol. 159, pp. 794–802, January 2019.

[29] A. E. Kah and I. Zeroual, “Arabic Topic Identification: A Decade Scoping Review,” *E3S Web of Conferences*, vol. 297, p. 01058, 2021.

[30] S. Khoja, “Shereen Khoja - Research,” 1999.

[31] K. Taghva, R. Elkhoury, and J. Coombs, “Arabic stemming without a root dictionary,” in *International Conference on Information Technology: Coding and Computing (ITCC’05)-Volume II*, vol. 1. IEEE, 2005, pp. 152–157.

[32] M. Alshomary, “light10stemmer,” 2015.

[33] K. Abainia, S. Ouamour, and H. Sayoud, “A novel robust Arabic light stemmer,” *Journal of Experimental &amp; Theoretical Artificial Intelligence*, vol. 29, no. 3, pp. 557–573, May 2017.

[34] A. Chelli, “Assem’s Arabic Stemmer,” 2018.

[35] T. Zerrouki, “Tashaphyne, Arabic light stemmer,” 2012.

[36] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, “Farasa: A Fast and Furious Segmenter for Arabic,” in *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 11–16.

[37] M. Masadeh, M. A, S. B, H. J, H. K, C. Chola, and A. Y. Muaad, “Investigating the impact of preprocessing techniques and representation models on arabic text classification using machine learning,” *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 1, 2024. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2024.01501110

[38] H. Tang, L. Zhou, X. Chengjie, and Q. Zhu, “A Method of Text Dimension Reduction Based on CHI and TF-IDF,” in *Proceedings of the 4th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering 2015*. Atlantis Press, 2015/12, pp. 1854–1857.

[39] M. Tubishat, M. A. M. Abushariah, N. Idris, and I. Aljarah, “Improved whale optimization algorithm for feature selection in Arabic sentiment analysis,” *Applied Intelligence*, vol. 49, no. 5, pp. 1688–1707, May 2019.

[40] K.-H. H. (Steeve), “Word2Vec and FastText Word Embedding with Gensim,” 2018.

[41] C. Fellbaum, M. Alkhalifa, W. Black, S. Elkateb, A. Pease, H. Rodriguez, and P. Vossen, “Introducing the arabic wordnet project,” in *Proceedings of the 3rd Global Wordnet Conference, Jeju Island, Korea, South Jeju, January 22-26, 2006*, P. Sojka, K.-S. Choi, C. Fellbaum, and P. Vossen, Eds., 2006, p. 295–299, proceedings of the 3rd Global Wordnet Conference, Jeju Island, Korea, South Jeju, January 22-26, 2006; Third Global Wordnet Conference ; Conference date: 22-01-2006 Through 26-01-2006.

[42] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, “AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP,” *Procedia Computer Science*, vol. 117, pp. 256–265, January 2017.

[43] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, “Advances in Pre-Training Distributed Word Representations,” in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018, pp. 52–55.

[44] N. Al-Twairesh, “The Evolution of Language Models Applied to Emotion Analysis of Arabic Tweets,” *Information*, vol. 12, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2078-2489/12/2/84

[45] M. M. Fouad, A. Mahany, N. Aljohani, R. A. Abbasi, and S.-U. Hassan, “ArWordVec: efficient word embedding models for Arabic tweets,” *Soft Computing*, vol. 24, no. 11, pp. 8061–8068, June 2020.

[46] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, “FastText.zip: Compressing text classification models,” December 2016.

[47] L. Abouenour, K. Bouzoubaa, and P. Rosso, “On the evaluation and improvement of Arabic WordNet coverage and usability,” *Language Resources and Evaluation*, vol. 47, no. 3, pp. 891–917, September 2013.

[48] V. Samawi, S. A. Yousif, and Z. Sultani, “Utilizing Arabic WordNet Relations in Arabic Text Classification: New Feature Selection Methods,” *IAENG International Journal of Computer Science*, vol. 46, pp. 750–761, November 2019.

[49] R. H. AlMahmoud and B. H. Hammo, “SEWAR: A corpus-based N-gram approach for extracting semantically-related words from Arabic medical corpus,” *Expert Systems with Applications*, vol. 238, p. 121767, 2024.

[50] B. Elayeb, “Arabic word sense disambiguation: a review,” *Artificial Intelligence Review*, vol. 52, no. 4, pp. 2475–2532, December 2019.

[51] T. I. Ababneh, S. M. Ramadan, and I. M. Abu-Shihab, “Perspectives on Arabic Semantics,” *International Journal of Humanities and Social Science*, vol. 7, no. 7, p. 8, 2017.

[52] S. Abdulateef, N. A. Khan, B. Chen, and X. Shang, “Multidocument Arabic Text Summarization Based on Clustering and Word2Vec to Reduce Redundancy,” *Information*, vol. 11, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/59

[53] R. Saidi and F. Jarray, “Sentence Transformers and DistilBERT for Arabic Word Sense Induction.” in *ICAART (3)*, 2023, pp. 1020–1027.

[54] R. Z. Al-Abdallah and A. T. Al-Taani, “Arabic Text Summarization using Firefly Algorithm,” in *2019 {Amity} {International} {Conference} on {Artificial} {Intelligence} ({AICAI})*, February 2019, pp. 61–65.

[55] R. H. AlMahmoud, B. Hammo, and H. Faris, “A modified bond energy algorithm with fuzzy merging and its application to Arabic text document clustering,” *Expert Systems with Applications*, vol. 159, p. 113598, November 2020.

[56] S. Al-Saqqa and G. Al-Naymat, *Unsupervised Sentiment Analysis Approach Based on Clustering for Arabic Text*. International Business Information Management Association (IBIMA), August 2020.

[57] A. A. Mohamed, “An effective dimension reduction algorithm for clustering Arabic text,” *Egyptian Informatics Journal*, vol. 21, no. 1, pp. 1–5, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866518301579

[58] A. R. Alharbi, M. Hijji, and A. Aljaedi, “Enhancing topic clustering for Arabic security news based on k-means and topic modelling,” *IET Networks*, vol. 10, no. 6, pp. 278–294, 2021, _eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/ntw2.12017.

[59] A. H. Aliwy, K. Aljanabi, and H. A. Alameen, “Arabic text clustering technique to improve information retrieval,” in *AIP Conference Proceedings*, vol. 2386, no. 1. AIP Publishing, 2022.

[60] R. H. Al Mahmoud, B. H. Hammo, and H. Faris, “Cluster-based ensemble learning model for improving sentiment classification of arabic documents,” *Natural Language Engineering*, pp. 1–39, 2023.

[61] S. Larabi-Marie-Sainte, M. Bin Alamir, and A. Alameer, “Arabic Text Clustering Using Self-Organizing Maps and Grey Wolf Optimization,” *Applied Sciences*, vol. 13, no. 18, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/18/10168

[62] R. H. AlMahmoud and M. Alian, “The effect of clustering algorithms on question answering,” *Expert Systems with Applications*, vol. 243, p. 122959, 2024.