Asia-Pacific Journal of Information Technology and Multimedia 12(1) 2023: 70 – 110
https://doi.org/10.17576/apjitm-2023-1201-05

Arabic Text Summarization using Pre-Processing Methodologies and Techniques

Penyarian Teks Arab menggunakan Kaedah dan Teknik Pra-Proses

Mohamed Yassin Abdelwahab, Yazeed Al Moaiad, Zainab Binti Abu Bakar

Al-Madinah International University, Taman Desa Petaling, 57100 Kuala Lumpur, Malaysia.

**Corresponding author: CO477@lms.mediu.edu.my*

Received 30 December 2022
Accepted 20 February 2023, Available online 1 June 2023

ABSTRACT

Recently, one of the problems that has arisen due to the amount of information and its availability on the web, is the increased need for effective and powerful tools to automatically summarize text. For English and European languages an intensive works has been done with high performance and nowadays they look forward to multi-document and multi-language summarization. However, Arabic language still suffers from the little attention and research done in this field. In our research we propose a model to automatically summarize Arabic text using text extraction. Various steps are involved in the approach: preprocessing text, extract set of features from sentences, classify sentence based on scoring method, ranking sentences and finally generate an extract summary. The main difference between our proposed system and other Arabic summarization systems are the consideration of semantics, entity objects such as names and places, and similarity factors in our proposed system. In recent years, text summarization has seen renewed interest, and has been experiencing an increasing number of research and products especially in English language. However, in Arabic language, little work and limited research have been done in this field. will be adopted Recall-Oriented Understudy for Gisting Evaluation (ROUGE) as an evaluation measure to examine our proposed technique and compare it with state-of-the-art methods. Finally, an experiment on the Essex Arabic Summaries Corpus (EASC) using the ROUGE-1 and ROUGE-2 metrics showed promising results in comparison with existing methods.

Keywords: Arabic text summarization; machine learning; natural language processing

ABSTRAK

Baru-baru ini, salah satu masalah yang timbul akibat banyaknya maklumat dan ketersediaannya di tapak sesawang adalah keperluan yang meningkat untuk alat yang berkesan dan berkuasa untuk merangkumi teks secara automatik. Untuk bahasa Inggeris dan bahasa-bahasa Eropah, telah dilakukan kerja intensif dengan prestasi yang tinggi dan pada masa kini mereka mencari

merangkumi ringkasan pelbagai dokumen dan bahasa. Walaupun begitu, bahasa Arab masih mengalami kurang perhatian dan kajian dalam bidang ini. Dalam kajian kami, kami mencadangkan sebuah model untuk merangkumi teks Arab secara automatik dengan menggunakan pengekstrakan teks. Pelbagai langkah terlibat dalam pendekatan ini: pra pemprosesan teks, pengekstrakan set ciri dari ayat, mengklasifikasikan ayat berdasarkan kaedah penilaian, penempatan ayat, dan akhirnya menghasilkan ringkasan pengekstrakan. Perbezaan utama antara sistem yang dicadangkan dan sistem-sistem merangkumi Arab yang lain adalah pertimbangan semantik, objek entiti seperti nama dan tempat, dan faktor keserupaan dalam sistem yang dicadangkan kami. Dalam beberapa tahun terakhir, merangkumi teks telah menarik minat baru, dan mengalami peningkatan jumlah kajian dan produk terutama dalam bahasa Inggeris. Walau bagaimanapun, dalam bahasa Arab, kajian yang sedikit dan terhad telah dilakukan dalam bidang ini. Kami akan mengambil Recall-Oriented Understudy for Gisting Evaluation (ROUGE) sebagai langkah penilaian untuk menguji teknik yang dicadangkan kami dan membandingkannya dengan kaedah-kaedah terkini. Akhirnya, satu eksperimen di atas Essex Arabic Summaries Corpus (EASC) menggunakan metrik ROUGE-1 dan ROUGE-2 menunjukkan hasil yang menjanjikan berbanding kaedah-kadah yang sedia ada.

Kata kunci: Penyarian teks Arab; pembelajaran mesin; pemprosesan bahasa semula jadi

INTRODUCTION

Automatic text summarization (ATS) is a technique designed to automatically extract salient information from related documents, which helps to produce a summarized document from a related set of documents (Abbasi-ghalehtaki, Khotanlou, & Esmaeilpour 2016; Zou et al. 2013). Nowadays, the amount of text data is increasing rapidly in areas such as news, official documents, and medical reports, so there is a need to compress such data using machine learning techniques, and text summarization can assist in extracting the significant sentences from various related documents (Sanchez-Gomez, Vega-Rodríguez, & Perez 2019). The main problems related to document summary are redundancy, noisy information, incoherency, and diminished readability (Verma & Om 2019).

Text Summarization is one of those applications of Natural Language Processing (NLP) which is bound to have a huge impact on our lives. One method that deals with natural language processing (NLP) is ATS, which extracts important sentences from related documents. Many researchers have focused on examining European languages and English at the Text Analysis Conference (TAC) and the Document Understanding Conference (DUC), however, there is a shortage of research on the Arabic language (Mallick et al. 2019). There are many types of methods that can classify text summarization, and summarization. Our research has examined multi document text summarization based on extracting related and significant information from the Arabic language within a generic context (Kanapala, Pal, & Pamula 2019). The main goal of this study is to introduce a domain specific document text summarization model and exploit them for the purpose of generating highly relevant summary sentences.

1. To investigate state-of-the-art of text summarization approaches proposed in Arabic language,
2. To analyze the limitation of current approaches and the peculiarity of Arabic language, which have posed challenge to the task of Arabic text summarization.

3. To propose the main lines of a new approach, which combines semantic information extracted from Arabic WordNet and rhetorical structure theory (RST), one of the most widely used discourse theories in natural language processing.

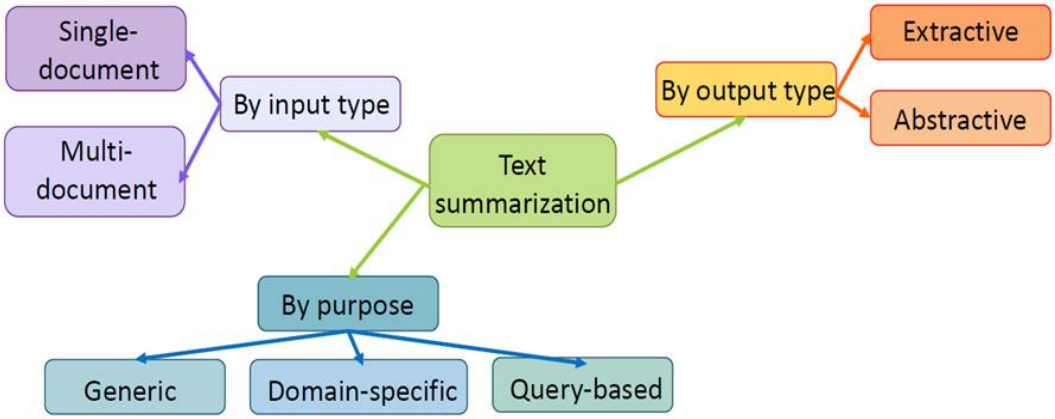

FIGURE 1. Process of text summarization taxonomy

Text summarization is one of the most important applications of Natural Language Processing (NLP). It is an essential tool for assisting and interpreting text information. The goal of automatic text summarization is to abbreviate one or more texts into shorter version conserving their information contents and overall meanings (Gupta & Lehal 2010). This will help the reader to decide if a document covers desired information with minimum effort and time loss.

Following the above objectives, we investigated extractive summarization, commonly used text summarization methods. We implemented extractive summarization using TextRank (Mihalcea & Tarau 2004) and TF-IDF algorithms (Ramos 2003).
Propose an unsupervised technique to deal with these problems that is based on combined multilevel features, such as important phrases, sentence similarity with titles, and sentence location.

Machine learning has become one of the most important topics within development organizations that are looking for innovative ways to leverage data assets to help the business gain a new level of understanding.

Automatically summarize Arabic text using text extraction. Various steps are involved in the approach:

1. preprocessing text,
2. extract set of features from sentences,
3. classify sentence based on scoring method,
4. ranking sentences,
5. extract summary.

Intelligent applications like Automated Text Summarization (ATS) can be developed to access this information.

Specialized Arabic ATS approaches are needed to overcome this problem and to support Arabic Natural Language Processing (NLP) systems.
Text summarization can be categorized into many types.

1. According to input factors, can generate single-document summary from a single input text document, or multi-document summary that is generated from multiple input documents concerning the same topic.
2. With regard to the output of the text summarization, a summarization could be extractive, that selects important sentences from the original document and concatenates them into a shorter form; or,
3. It could be Abstractive, which attempts to understand the main concepts in the document and to express them in a clear language through representing them using new terms, expressing them using new formulations, and mentioning other concepts and words than those in the original text.

PROBLEM BACKGROUND

Arabic has characteristics that make it more beautiful and more expressive, but this the process of linguistic analysis and building applications that can process and summarize the language is difficult, and these are some of the characteristics that are more present in the Arabic language than the English language:

1. Orthographic Ambiguity & Inconsistency
2. Morphological inflections
3. Word order freedom
4. Dialectal variation
5. Arabic Script & Encoding Problems
6. Phonology & Spelling
7. A great variety of Arabized names
8. The use of the Latin alphabet to spell the Arabic language, and it is without fixed rules.
9. The intuition of the person who writes and suffers from previous problems more clearly.

CHALLENGES

The main challenge in Arabic text summarization is in the complexity of the Arabic language itself:

1. The meaning of a text is highly dependent on the context.
2. There are more inherent variations within Arabic than any other language.
3. The diacritics are usually absent in the texts of news articles and any online content.

The Arabic language has 28 characters and each character's shape changes based on its position, e.g., the letter "ح" has three different shapes: "حـ" at the beginning of the word; "ـحـ" in the middle; and "ـح" at the end.

Arabic has two types of vowels: long vowels represented by letters and short vowels appearing as diacritical marks.

1. The size of the Arabic alphabet can be extended to ninety.
2. All words in Arabic are derived from a list of roots with 3 or 4 constants.
3. Morphological analysis is hard because the language is derivational and inflectional.

THE CHALLENGE OF THE REDUNDANCY

The problem for Arabic multi document summarization is formulated as follows:
Given a set of documents, MD, that is, MD = (D1, D2, : : : : : : ., Dn) where Di indicates the ith document in MD, n represents the fold document in all text, then, we tokenize the original document D to a list of sentences, that is, D = (S1, : : : , Sn), where Si represents the ith sentence in D, and n represents the total number of sentences in each document. The goal of the final summary is to select the set of sentences from MD covering various related topics from related documents.

Natural language processing of the Arabic language is challenging and has the following key properties:

1. The Arabic language is diacritical and derivative, making morphology analysis a hard task.
2. Arabic words are often imprecise since the system is based on a tri-literal root.
3. Broken plurals, where a broken plural in linguistics is an irregular plural form of a noun or adjective initiate in the Semitic Arabic languages.
4. Characters can be written in different ways based on the location of the character in a word.

The main idea for this Research is to reduce the redundancy issue by focusing on extracting key sentences. These sentences should contain the main idea for correlated documents for making a comparison between two sentences. One of these sentences is considered redundant if the similarity between these two sentences is high (based on a chosen threshold of similarity), therefore, only one of the associated sentences is selected thereafter.

PROBLEM STATEMENT

With the increase in the number of articles and texts issued daily, the need for summarization increased in order to be able to absorb this amount of information and evaluate what is important and not important to us. Therefore, the main challenge is the difficulty of the Arab community in summarizing Arabic texts with the lack of linguistics, high grammar, semantics, eloquence, and the difference in dialects among Arab peoples, which resulted in incomplete text and lack of understanding of the context of the meaning to be analyzed in order to take peaceful decisions based on the context and understanding the meaning.

RELATED WORK

Many research were conducted in the Arabic text summarization. These researchers used different techniques and algorithms as presented in the following subsections. Summarization models may be classified into extractive summary and abstractive summary. Extractive summarization model creates extracts by selecting important sentences and another creates abstract by understanding the meaning of the whole text. In this research, extractive method is focused, that is, those which select sentences from the original document to produce the summary. LexRank, is a centroid-based method for multi-document summarization that computes the sentence importance based on the concept of eigenvector centrality. It assumes a fully connected, undirected graph with sentences as nodes using tf-idf score and similarities between them as edges using cosine similarity. A very

similar system is TextRank, has also been proposed for single-document summarization. The following proposed algorithm is different from each of these and computes the semantic distance between the sentences in the document and then applies TextRank algorithm. Most of the existing summarization does not use semantic content of the sentence and relative importance of the content to the semantic of the text. The proposed approach is based on identifying the semantic relations among sentences.

Deep learning, an area in machine learning, has performed with state-of-the-art results for common NLP tasks such as Named Entity Recognition (NER), Part of Speech (POS) tagging or sentiment analysis (Zou et al. 2013). In the case of text summarization, the two common approaches are extractive and abstractive summarization. For extractive summarization, dominant techniques including TF-IDF and TextRank (Hasan & Ng 2010). TextRank was first introduced by Mihalcea, Rada, and Paul Tarau in their paper TextRank: Bringing order to text. The paper proposed the idea of using a graph-based algorithm similar to Google's PageRank to find the most important sentences. Juan Ramos proposed TF-IDF.

Abstractive summarization is most commonly performed with deep learning models. One such model that has been gaining popularity is sequence to sequence model (Nallapati, Zhou, & dos Santos 2016). Sequence to sequence models have been successful in speech recognition and machine translation (Sutskever, Vinyals, & Le 2014). Recent studies on abstractive summarization have shown that sequence to sequence models using encoders and decoders beat other traditional ways of summarizing text. The encoder part encodes the input document to a fixed-length vector. Then the decoder part takes the fixed-length vector and decodes it to the expected output (Bahdanau, Cho, & Bengio 2014).

TABLE 1. Difference between extractive and abstractive text summarization techniques

	Extractive Automatic Text Summarization Techniques		Abstractive Automatic Text Summarization Techniques
1.	Statistical-Based Methods	1.	Methods Based on Graphs
2.	Graph-Based Methods	2.	Methods Based on Trees
3.	Semantic-Based Methods	3.	Methods Based on Rules
4.	Machine-Learning Approaches	4.	Methods Based on Ontologies
5.	Fuzzy-Logic Approaches	5.	Methods Based on Semantics
		6.	Deep-Learning-Based Approaches

EXTRACTIVE AUTOMATIC TEXT SUMMARIZATION TECHNIQUES

Statistical-Based Methods

Summarizing text can also be done using various statistical methods. Sentence selection may be based on the set of features which depend on inverse document frequency (IDF), term frequency (TF), similarity with the document title, binary term occurrences, and term occurrences. This method is simple to execute and can be used to eliminate redundancy. In other words, there are various approaches that are based on statistical methods that can obtain a set of features in order

to improve the final results, like the approach used in (Waheeb et al. 2020), which improved results when used in combination with statistical and other methods. This method was based on combining sentence location and semantic score, and it examined a single document of Arabic language (Essex Arabic Summaries Corpus (EASC) Corpus), achieving a F-measure of 0.57. For solving the redundancy problem, we applied statistical-based methods to improve the selection of significant sentences. In addition, the statistical methods enhance the results when combined with other methods, also known as a hybrid approach (Kanapala et al. 2019).

Graph-Based Methods

In these methods, text data are shown as a graph, where the nodes of the graph represent the sentences, while the edges among the nodes represent similarity relationships among sentences. The approach presented by (Wei et al. 2010) applied multi document sensitive ranking. Their method highlights the effect of the set of documents as global information for evaluating local sentences based on the document-to-document relations and document-to-sentence relations. a graph-based method to relate sentences to documents, where these relationships were calculated by a graph-based ranking algorithm (Wan & Yang 2008). This method is based on a graph model that assigns relationships between different sets of documents and sentences, where it then examines the sentence evaluations based on the sets of entire document relationships.

The graph document model was combined to identify the document impact by determining sentence to document relationships and document importance for ranking the sentences. a multimedia summary technique based on an online social media network for generating multimedia stories. Their research focused on the sharing and management of multimedia data on a social media website, and it focused on the influence of analysis methodologies and graph-based models for discovering the most significant data that was related to one hot topic. They modified their Artificial Bee Colony (ABC) algorithm for ranking, selection, and the semantic correlation between two objects (mixture of texts and pictures). The experiments were conducted using the YFCC100M dataset, and the ROUGE-2 and ROUGE-SU4 metrics were used to test the evaluation (Amato et al. 2018). With graph-based approach, the document is represented in the form of undirected graph. There is a node for every sentence. An edge between two nodes is drawn if there is a relation between these two nodes. A relation can be a cosine similarity above a threshold, or any other type of relationship. After drawing a graph, it is possible to view the sub-graphs of connected nodes as a cluster of distinct topics covered in the document. This yields two results: For query-specific summaries, sentences may be selected only from the pertinent sub-graph, while for generic summaries sentences would be selected from each sub graph for best coverage.

Recently (Al-Taani & Al-Omour 2014), we are representing each document by weighted directed graph whose nodes represent sentences and edges weights represent similarity between sentences. This similarity is determined by ranking the sentences according to some statistical features. The cosine similarity measure is chosen based on term weighting scheme, which is the TF-IDF (Term Frequency-Inverse Document Frequency). The summary is extracted by finding the shortest path between the first and the last nodes in the graph considering the user compression ratio.

Semantic-Based Methods

This method focuses on extracting the relationships between the words based on the semantics of the words. These kinds of techniques can be used for text summarization problems also, however, the drawback of these methods is that they need a specific tool to achieve a high-quality summary, such as in linguistic resources and semantic analysis (Lexical, WordNet). This approach requires intensive memory to store semantic relationships and also requires high-performance processors for complex linguistic processing, semantic knowledge, and additional linguistics. The words' meaning can be presented using the semantic relationship between terms and sentences for providing useful information from the text corpus.

Latent Semantic Analysis (LSA) is a completely unsupervised approach for learning and capturing the contextual use and meaning of words using statistical calculations. By utilizing the semantic content of words, it avoids the issue of synonymy.

1. Using Singular Value Decomposition (SVD) can find principal orthogonal dimensions of multidimensional data. It is named LSA because SVD applied to document word matrices, groups documents that are semantically related to each other.
2. Using Singular Value Decomposition (SVD) can find principal orthogonal dimensions of multidimensional data.

In singular value decomposition, the input matrix is decomposed into three other matrices to model the relationship between words and sentences.

1. The first and third matrices represent the vector of extracted values for the original rows and original columns, respectively.
2. The second matrix represents scaling values and the third matrix represents original columns as the vector of extracted values.

LSA is composed of three main steps:

1. The generation of an input matrix,
2. The use of singular value decomposition (SVD), and
3. Sentence selection

Machine-Learning Approaches

This approach, which has been applied in binary classification with multi-document summarization and has shown promising results, requires labeled data to train the model. In other words, the performance is affected by selecting a significant set of features, and the representation of these features plays an important role in this approach's performance. With this method, sentences are transferred into vectors. These vectors are calculated for different levels of features, for example, tokening the documents to paragraphs, sentences, or words, and frequency is calculated to extract the relationship between them (Belkebir & Guessoum 2015).

Support Vector Machine (SVM) and AdaBoost algorithms for Arabic document summarization based on machine learning. This method decides which sentences will be selected for the final summary. The first step of this method is to apply two classifier techniques, namely, SVM and AdaBoost. The second step is predicting the sentence for the final summary by the AdaBoost and

SVM classifiers. The performance of machine learning is affected by the selected classifier method, the set of features selected, and the set of features represented, which play a significant role in the performance of this method (Amato et al., 2019).

In machine learning based approach, the summarization process is modeled as a classification problem: given a set of training document and their extractive summaries, each sentence is classified as a summary sentence or non-summary sentence based on statistical features. Some Arabic summarization systems have been adopting machine learning and statistical techniques. For instance, (Sobh, Darwish, & Fayek 2009) integrate Bayesian and genetic programming classification methods in an optimized way to extract the summary sentence using reduced feature set. The system requires training and uses manually labeled corpora. Experiments show that Bayesian classifier tends to have large recall unlike GP classifier, which tends to have large precision. By integrating both classifiers, the researcher found that using the union for integration increased the recall and the summary size, while using the intersection for integration increased the precision and decreased the size of the summary. Later, (Fattah & Ren 2009) investigate the use of genetic algorithm (GA), mathematical regression (MR), feed forward neural network (FFNN), probabilistic neural network (PNN) and Gaussian mixture model (GMM) for automatic text summarization task. Ten features are used in combination to train the over mentioned methods on a manually created corpus. Numerous experiments were performed. In addition, results indicated that GMM model is the best.

Fuzzy-Logic Approaches

Fuzzy Logic rule and fuzzy logic set are used to extract the important sentences based on their features. The techniques provide decision-support and expert systems with strong reasoning capabilities.

Eight features for text summarization:

1. Title word,
2. Sentence length,
3. Sentence position,
4. Numerical data,
5. Thematic words,
6. Sentence to sentence similarity, and
7. Term weight and
8. Proper Nouns.

The system involves in following steps:

1. In the preprocessing step, the system extracts the individual sentences of the original documents. After, separate the input document into individual words.
2. The features are calculated to obtain the sentence score based on fuzzy logic method.
3. A set of highest score sentences are extracted as document summary based on the compression rate.

This method considers each feature of a text such as sentence length, similarity to key word and others as the input of fuzzy system. Then, it develops and enters all the rules required for summarization, in the knowledge base of system. then, a value between zero to one is obtained for each sentence in the output based on sentence characteristics and the available rules in the knowledge base. The input membership function for each feature is divided into three membership functions which are composed of insignificant values (low L), very low (VL), medium (M), significant values (High h) and very high (VH). The important sentences are extracted using IF-THEN rules according to the feature criteria. text summarization based on fuzzy logic system architecture design usually implicates selecting fuzzy rules and membership function. the selection of fuzzy rules and membership functions directly affect the performance of the fuzzy logic system.

ABSTRACTIVE AUTOMATIC TEXT SUMMARIZATION TECHNIQUES

Methods Based on Graphs

1. Each node acts as a word, and nodes are linked by positional information.
2. The structure of sentences is represented by coordinated edges.
3. The graph-based method's preparation processes include graph creation, constructing a textual graph to represent the original text, and summary creation.
4. It may be used in any domain and does not need the involvement of human expertise.
5. By connecting all words on a word graph route, a new phrase is created.
6. The disadvantage of this strategy is that word charts do not represent the meaning of the words.

Methods Based on Trees

1. These algorithms find similar comparison statements and combine them to generate the abstractive summary.
2. Similar sentences are represented by a tree. Dependency trees are the most frequently used tree-form representations for text.
3. Trees are managed through pruning, linearization (converting trees to strings), and other methods.
4. The method's advantages include enhanced quality generated summaries since language generators provide fewer redundant and fluent summaries.
5. It is not feasible to discern relationships between sentences without first locating common terms.
6. Because it ignores the context, it misses a variety of important phrases within the material.
7. The effectiveness of this technique is limited by the available parsers.

Methods Based on Rules

1. These methods need to establish the rules and categories in order to determine the most essential ideas in the input text.
2. This method's stages are as follows:

3. To construct an abstractive summary, one must first categories the input text based on words.
4. The generated summaries are high in information.

The ability to handle additional data is by increasing abstraction scheme complexity and variety.

Methods Based on Ontologies

1. Each domain has its own set of articles, each with its own information structure.
2. Data arrangement an ontology such as may be expressed.
3. The basic idea is to utilize an ontology to extract relevant information from a text and construct an abstract summary.
4. It is based on publications from a certain domain.
5. It can deal with text uncertainty and provide logical summaries.

Methods Based on Semantics

Traditional dialect period frameworks are utilized as verb and noun phrases to generate the final abstractive summary.

The propose a multi-document abstractive summarizer that:

1. Utilizes Semantic Role Labeling (SRL) to talk to input documents.
2. Uses Semantic Role Labeling (SRL) to communicate with output documents.
3. Cluster semantically identical predicate-argument structures throughout the content
4. Order the predicate-argument structures based on the semantic proximity metric.
5. The Semantic Role Labeling (SRL) may be used to bind words together.
6. The quality of the output summary is determined by the input text's semantic representation.

Deep-Learning-Based Approaches

Sequence-to-Sequence learning (seq2seq) has made abstract summarization possible. It has been used successfully in NLP applications such as machine translation, speech recognition, and conversation systems.

Deep learning still has concerns with

1. Creating repeated words or phrases and
2. Not dealing with terms that are not in the vocabulary.

RNN models use attention encoder-decoder to summarize material effectively.

Deep learning approaches still suffer from challenges such as:

1. Creating repeated words or phrases and
2. The inability to cope with words out of vocabulary (OOV) (i.e., unusual and limited occurrence words).

The summarizing mechanism of goes like this:

1. Separate the actual items (for example, news reports) and their summaries.
2. After preparing the data with a sub-word display, do word segmentation.

3. Using a pre-trained Genism toolkit to initialize the word vectors, with one layer of Bilstm for the encoder and a unidirectional.

LSTM layer for the decoder. The cost function was optimized using the Adam optimizer (loss).

Deep learning models are typically employed for brief text summaries.

Combining multiple approaches and tactics are advised to develop improved abstractive summaries.

It is very promising to combine results from numerous ATS techniques to provide considerably better summaries than those created by individual algorithms.

These methods might be employed in preprocessing to extract key terms from the input text and then utilized to generate the abstractive summary.

THE PROPOSED APPROACH

The propose of a graph based Arabic ATS approach, which focus on the semantic relationships between the sentences, in this approach a single document is represented by a graph, where sentences are represented by the nodes of the graph, and the similarity between each two sentences is represented by the weight of the edge between each two nodes. The summary is extracted by finding the shortest path between the first sentence in the original document and the last sentence (first and last nodes in the graph), the nodes that form the shortest path are the extracted summary. Statistical features are determined to rank the sentences, such as: sentence length, sentence position, term frequency and title similarity.

PageRank formula is used to combine both ranking sentences and calculating similarity between sentences. To calculate the similarity measure and sentence score, first, we determine the basic unit on which these calculations are based. In this study, three basic units are applied; stem, word, and n-gram. Differences in calculating similarity measure, sentence ranking, and the quality of the extracted summary are examined according to each unit.

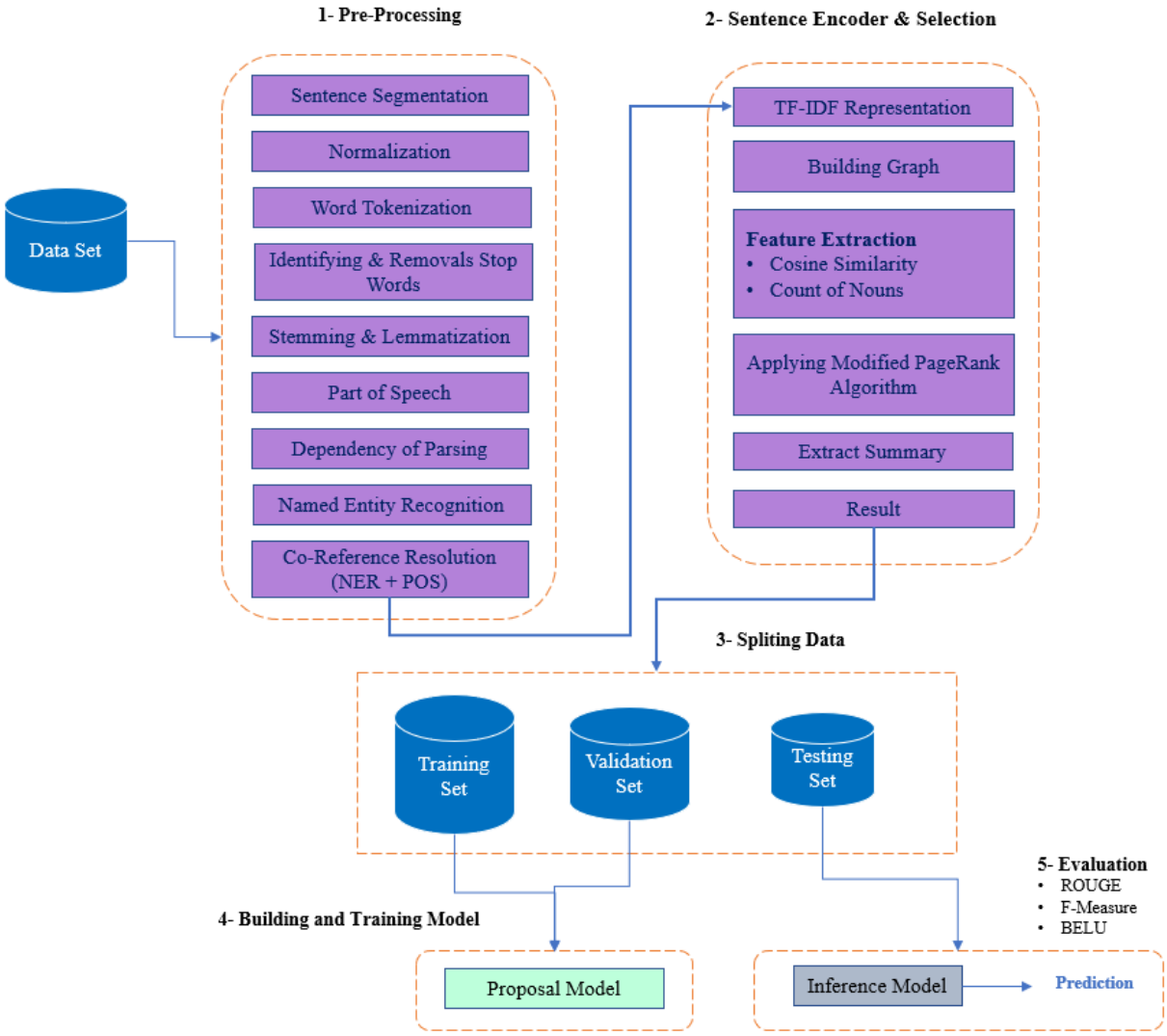

FIGURE 2. Proposed approach

Input Single Document

In this step, the system reads a single document written in Arabic with the file extension (.txt). Encoding format is taken into consideration and should be compatible with the Arabic language.

Preprocessing

The process of constructing the NLP system is not straightforward due to the nature of the Arabic language. Arabic is well-known for its rich and complex morphological and syntactic flexibility. Therefore, some language processes, such as tokenization, stemming, and normalization should take place before starting summarization steps. At the beginning, the system deals with the Arabic texts without diacritical marks since most texts which are written in Arabic and saved in an electronic form do not have these marks.

This stage is the initial stage in almost all summary methods. Its main purpose is to prepare the input text document for processing in other stages. It mainly transforms the input document into a unified representation. The proposed text summary system includes the following preprocessing sequenced operations: tokenization, letters normalization, stop-words removal, and stemming (Aries, Zegour, & Hidouci 2015).

Normalization

This step has a great effect on the quality of the extracted summary as the redundant and misplaced white spaces are corrected. Furthermore, any consecutive punctuation marks are removed. Also, if there is a repeated sentence in the text, it is removed after splitting the document into sentences. This is a significant step because one of the major merits of an efficient summary is that it should not contain any redundant data. To avoid this problem, any repeated information in the single document is removed.

In the Arabic language, some Arabic letters might appear in different forms, while other characters are used instead of others because their shapes are similar. Moreover, writers use diacritics in their texts. These create a set of variations for the same term; and thus, affect the computation of some features such as Term Frequency (TF). Therefore, a normalization process is required to unify the different forms of the same letter to avoid such variations. The proposed normalization step employs AraNLP tool to do the following tasks (Althobaiti, Kruschwitz, & Poesio 2014):

1. Removing non-Arabic letters such as special symbols and punctuations,
2. Removing diacritics,
3. Replacing آ – إ – أ with ي – ا , with ي – ى , and ة - with ه (Ayedh, Tan, Alwesabi, & Rajeh, 2016). And
4. Removing tatweel (stretching character).

Tokenization

Tokenization is a major problem in text summarization since it deals closely with the morphological analysis of documents, particularly those which are written in languages of rich and complex morphology, such as Arabic. The function of a tokenizer is to split a running text into tokens, so that they can be fed into a morphological transducer or Part of Speech (POS) tagger for further processing. Also, the tokenizer is responsible for the determination of word boundaries and the demarcation of clitics, multiword expressions, abbreviations, and numbers. In this step, sentences are divided into words on the basis of white spaces and punctuation marks in preparation for the processes of later steps like stemmer and POS tagging.

Text preprocessing starts with the tokenization process which splits the input documents into their units with different levels to facilitate accessing all parts of the input document. These units are paragraphs, sentences, tokens, numbers, or any other appropriate unit. To illustrate, the proposed tokenization is a morphological decomposition based on punctuation which starts with finding the paragraphs that the document consists of, where the newline character (\n) is the paragraph delimiter. After that paragraphs are split into a set of sentences based on full stop (.), question mark (?), and exclamation mark(!) as delimiters. Finally, these sentences are divided into tokens based on delimiters like white space, semicolons, commas, and quotes (Althobaiti et al. 2014).

Stop Word Removal

Stop words are the words which are excluded before the automatic language processing of texts. They are repeated in the texts such as (...، من الى ،في) and it is better ignored and not indexed so that the search could be improved.

Stop words have two major effects on some applications like text summarization and information retrieval. They could influence the efficiency of retrieval as such frequent words have a high tendency to diminish the differences in frequency among less common words affecting the weighting process. Also, stop words removal shortens the length of the document and, consequently, affects the weighting process. Efficiency can be affected by such removal because such stop words are meaningless and not functional.

In this research we used (El-Khair 2006) stop list which contains 1,377 words. The list consists of the following word categories: Adverbs, Conditional Pronouns, Interrogative Pronouns, Prepositions, Pronouns, Referral Names/ Determiners, Relative Pronouns, Transformers (verbs, letters), Verbal Pronouns, and others.

Stop words (i.e. pronouns, prepositions, conjunctions, etc.) are insignificant words that frequently appear in the documents to form sentences (Khan & Salim 2014). Since these words are not informative (do not add information), they can be eliminated from sentences without affecting the core content of the sentence. Indeed, this step is crucial since some of the calculations are based on the words' frequencies in the sentence/document. Thus, by removing stop words, these calculations become more relevant and accurate. There are several stop-list methods that are used to remove stop-words from the text including, General Stop-list, Corpus-Based Stop-list, and Combined Stop-list.

Stemming

Arabic is a highly inflectional and derivational language, which means that Arabic words can have many different forms but share the same abstract meaning of action. This has, evidently, affected many natural languages processing methods such as building bag-of-word model and text similarity calculation. Therefore, Stemming is the process of removing some or all affixes (e.g. prefixes, infixes, postfixes, and suffixes) from a word. In other words, stemming transforms the different forms/derivatives of a word to a single unified form (e.g. root or stem) from which all the derivatives are generated. In Arabic, there are two common stemming approaches; Morphological root-based stemming and light stemming (Mustafa et al. 2017).

Stemming is the process of reducing words to their roots through the removal of any affixes attached to them. For example, stemming the Arabic word "الكتابة" produces the root "كتب". This root can also be generated from the word "كاتب". After reducing words to their roots, these generated roots can be used for many applications like compression, spell checking, and text searching.

In this proposed approach, the stem of the word is employed as the basic unit for calculating the similarity measure in addition to POS tagging, which is performed in a later step.

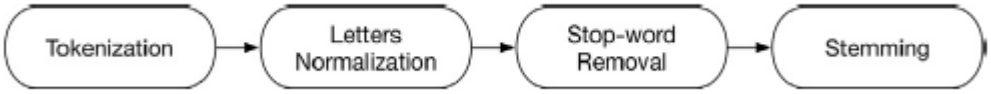

FIGURE 3. Sequence of the preprocessing steps

Part-Of-Speech Tagging

Arabic language includes three major parts of speech: nouns, verbs, and particles. The POS tagger assigns the suitable POS for each word. The tagging system consists of three main levels: morphological, lexical, and syntactic analysis.

TABLE 2. Part of speech tagging example

الرومانيين بيقسمو الكاربات فى بلدهم لتلات مجموعات: الكاربات الجنوبية والكاربات الغربية والكاربات الشرقية										
الرومانيون	بيقسمو	الكاربات	في	بلدهم	لتلات	مجموعات	:	الكاربات	الجنوبية	و
noun	verb	noun	prepositions	Noun + pronoun	Preposition + word after preposition	noun	Stop words	noun	noun	Stop words

Noun Extraction from Sentences

Not all words in a document are good indicators of key phrases which reflect the significance of the sentences and the possibility of presence in the summary. Therefore, the process of syntactic filters is applied in this phase, and only nouns are extracted. These extracted nouns are the basis of sentence weight calculations.

Text Extraction

Extractive summarization means extracting keywords or key sentences from the original document without changing the sentences. Then, these extracted sentences can be used to form a summary of the document.

Textrank

TextRank is an algorithm inspired by Google's PageRank algorithm that helps identify key sentences from a passage (Mihalcea & Tarau 2004).

Using this idea, one can create a graph of sentences connected with all the similar sentences and run Google's PageRank algorithm on it to find the most important sentences. These sentences would then be used to create the summary. TextRank is an extractive and unsupervised text summarization technique. Let's take a look at the flow of the TextRank algorithm that we will be following:

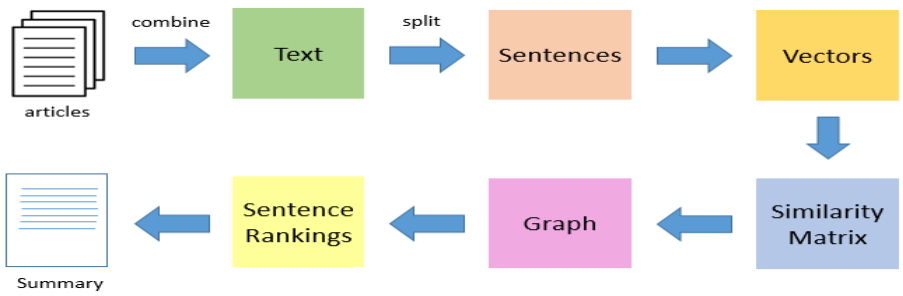

FIGURE 4. TextRank Flow Chart

PageRank Algorithm

The internet is a very complex network. The relation between pages can be represented as a graph. The importance of any vertex is the in-degree (out-degree); which is the number of inbound (outbound) links to this vertex (Bahdanau et al. 2014). The inbound of a given page, is an indicator of a page's importance or quality. PageRank algorithm (Bahdanau et al., 2014) uses this idea to rank the pages that appear in the search results. PageRank does not consider all the inbound links from the pages equal, the link will take an extra importance depending on the importance of the page that comes from it.

Algorithm 1. PageRank algorithm

Input: Weighted Graph G.

Output: Scored Graph.

1. Configure N = Number of Nodes in G.
2. Current_Rank Double[N]
3. Temp_Rank Double[N]
4. Foreach n = 1 to N
5. Current_Rank[n] = 1/N
6. For i = 1: Number_of_Iterations
7. Foreach nd: G. Nodes
8. Temp_Rank [nd. index] = Calc_Page_Rank(nd)
9. Current_Rank = Temp_Rank

The PageRank algorithm (Bahdanau et al. 2014) of a Web page u, denoted by PR (u), as shown in Formula where d is the damping factor with 0.85 and N: is the total number of nodes. Algorithm 1 shows how the PageRank algorithm works; it starts by initializing the rank of each node by 1/N, where N is the number of nodes in the graph. Then the algorithm iterates and calculates the new ranks of the nodes according to Formula. After calculating the new ranks for all nodes, the ranks are updated. This process is repeated depending on iterations count. The iteration here is used to update the rank for each sentence to get the best and most stable rank, since the order of the sentence is directly associated with the order of the associated sentences, which changes every time the algorithm is applied.

$$PR(P) = (1 - d) + d * \sum_{i-1}^{N} \frac{PR(Vi)}{N(Vi)} \quad (1)$$

Modified PageRank Algorithm

Nouns in Arabic language have a special importance; i.e. the more nouns a sentence has the more important it becomes (Bahdanau et al. 2014). So, this research uses a new technique that uses the number of nouns in each sentence to modify the original PageRank algorithm to extract Arabic summaries. A Modified PageRank is based mainly on the aspects of PageRank algorithm with the following differences:

1. Pages are replaced with the sentences of the document,
2. The weight of the edges between nodes calculated by the cosine similarity, while in the original PageRank there is no weight on the edges.
3. The initial rank of each sentence is the number of nouns in this sentence not like the original PageRank which gives the initial rank equally to all nodes which equals 1/N, Where N is the number of sentences in the document.
4. The PR (vi) is modified as in Formula. Formula is used to calculate the new rank of node (g), Where PR(vi) is the current rank of sentence (vi) and E(g, vi) is the weight of edge connect sentences (g) and (vi) which is also the cosine similarity between these two sentences, finally the summation is divided by the number of the remaining sentences in the document N-1, which is the number of sentences in the document D with excluding the current sentence, to get the new rank of sentence (g).

$$MPR(P) = (1 - d) + d * \sum_{i-1}^{N} \frac{PR(Vi) * E(g, Vi)}{N - 1} \quad (2)$$

TABLE 3. Modified PageRank (sentences initial rank)

Sentence #	S1	S2	S3	S4
Initial Rank (# of Nouns)	3	8	5	6

TABLE 4. Modified PageRank (edges weights)

Sentence #	S1	S2	S3	S4
S1	-	3	4	1
S2	3	-	6	5
S3	4	6	-	3
S4	1	5	3	-

TABLE 5. Modified PageRank (ranks after iteration # 1)

Sentence #	S1	S2	S3	S4
New Ranks after Iteration #1	14.31	19.7	22.25	16.58

Assume a document D that contains 4 sentences, and the number of nouns in each sentence presented in Table 1, the cosine similarity or the edge weights listed in Table 2. Then Table 3 shows the first iteration of calculating the new ranks of the sentence by considering that the value of the damping factor d here is 0.85 as best practice (Bahdanau et al. 2014). So, the Modified PageRank for (S1) calculated as in the following:

$$MPR(S1) = \frac{PR(S2) * E(S1,S2) + PR(S3) * E(S1,S3) + PR(S4) * E(S1,S4)}{3} \quad (3)$$

MPR(S1) = (8 * 3 + 5*4 + 6 * 1)/3 = (24 + 20 + 6)/3 = 50/3 = 16.67

New Ranks after Iteration #1 = (1-d) + d * MPR(S1)

New Ranks after Iteration #1 = (1- 0.85) + 0.85 * 16.67

New Ranks after Iteration #1 = 14:31

Table 6 shows the new ranks for the sentences after applying Modified PageRank for one iteration.

TABLE 6. Example of Arabic single document

S1	سلسة جبال الكارپات هي سلسة جبال في شرق اوروبا بتمتد في 7 دول أوروبية بتبتدى في (براتيسلافا) سلوفاكيا وبتعدى على بولاندا وتشيكيا واوكرانيا والمجر وصربيا وبتنتهى عند (اورشوفا) رومانيا عند نهر الدانوب، على شكل هلال طوله 1,500 كيلو متر وبتربط بين جبال الالب فى وسط اوروبا وجبال البالكان فى جنوبها.
S2	جبال الكاربات مشهوره بالمنتجعات السياحيه والرياضات الشتوية وبينبع منها انهار كتير اهمها نهر الدنيستير ونهر الفيستيولا ومصدر مهم للمعادن زي الدهب والفضة والرصاص والحديد وكمان فيها غابات الصنوبر والبلوط والزان وفيها تنوع كبير للحيوانات من اول الدببة والديابه والقطط البرية والغزلان.
S3	أكبر جزء من جبال الكاربات موجود فى رومانيا وبيتقال عليها الكاربات الرومانية حوالي 55.2% من جبال الكاربات موجود فى رومانيا و47.4% من أراضي رومانيا بتقع ضمن سلسلة جبال الكاربات.
S4	الرومانيين بيقسمو الكاربات فى بلدهم لتلات مجموعات: الكاربات الجنوبية والكاربات الغربية والكاربات الشرقية.

TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) is used to determine the relevance of a word in the document (Ramos 2003). The underlying algorithm calculates the frequency of the word in the document (term frequency) and multiplies it by the logarithmic function of the number of documents containing that word over the total number of documents in the dataset (inverse document frequency). Using the relevance of each word, one can compute the relevance of each sentence. Assuming that the most relevant sentences are the most important sentences, these sentences can then be used to form a summary of the document.

It depends on two important values:

1. Term Frequency The rate of repetition of the desired word
2. Inverse Document Frequency It indicates how rare or common this word is.

The second value indicates how common the word is, and it is inversely proportional to the strength of the word. Note that if the word itself is widespread and common among all texts, and its use increases in all other documents, this makes it have a weak effect. That is, any word that is spread

in many texts and documents with different meanings, this word is often less valuable and has less impact on the meaning. The words (very, yes, no, increase, decrease,) are common words with a general meaning and it will not be it has the effect of determining the type of text. While the specific words that are rarely found except in certain meanings such as (investment, benefits, bacteria, programming, grease), these words are found only in a limited number of documents, and therefore have a stronger meaning. The words with the most repetition in the files, may have a greater impact Indeed, but in the case when this word is basically rarely used.

A word such as: gold, is a word that is not widespread in general speech, and therefore if it is used a number of times in a file, it means something positive, while a word such as: indeed, is a word that is widespread in many and many files, and therefore if it is used a lot in a file, it doesn't mean anything specific. Thus, we want to calculate the extent of widespread and widespread use of words. Well-known words have less weight, and less-used words have a higher weight.

How can it be calculated?

$$idf_t = log_{10}\left(\frac{N}{df_t}\right) \quad (4)$$

This is done by calculating two numbers.

First: N which is the total number of files we have.

Second: The df number, which is the number of files in which the required word was mentioned.

And the model is called the inverse number of idf files because it depends inversely on the spread of the word in the files, so we divide N by df, and we make a logarithm for it, and if the word is rarely used, and it appears in a few documents, the logarithm will be a large number (million For example) and it has a large value of 6, but if it appears in a large number of documents and let it be half of them, it will be logarithm of 2, which is a small number, and if it appears in all files, it will be logarithm of 1 which is 0.

One of the uses of this matter is to determine the importance of search words. If we had a sentence like “try to buy an insurance” then before Google starts searching, it must determine which words have greater weight. If both words appear in certain files 10,000 times, but the word insurance had appeared in 4 thousand files, and the word try in 8 thousand, this means that insurance is less widespread, as it is more important and stronger in weight. So, the equation for calculating the strength of a word depends on the increase in its presence, but also on the decrease in the number of documents in it, so its name is term frequency, i.e. the frequency of the word, but inverse document frequency, i.e. the inverse of the number of documents, or by dividing it, and therefore the IDF equation is:

$$DF\ (term) = \frac{d(Number\ of\ documents\ containing\ a\ given\ term)}{D(the\ size\ of\ the\ collection\ of\ documents)} \quad (5)$$

$$IDF\ (term) = log\ [\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ a\ given\ term\ in\ it}] \quad (6)$$

Sometimes the number 1 is added to the fraction, to avoid getting a value that is log 0, and sometimes the number 1 is added to the fraction, to avoid getting the value of log1 which is 0. If the word is spread in all documents, the fraction is equal to 1, and the value of the IDF is log is log2 = 0.3 But if it is present in half the number of documents, the value is 0.47, and if it is present in 1% of the documents, the value is 2, and in the end the value of TF is multiplied by the IDF:

The final equation to combine TF and IDF, and final formula for combining both features is:

$$W_{t,d} = (1 + 1 + log\ tf_{t,d}) \times log_{10}(\frac{N}{df_t}) \quad (7)$$

It is based on the fact that the word has more weight when:

1. Its presence in the file increases.
2. It is less common in other files.

The TF-IDF tool is also used to initialize the text data before making a prediction or classification algorithm, through the following steps:

1. List all the words in the file.
2. Determine the extent to which each word is present in the chosen sentence.
3. Calculate the value of tf-idf and put it in its exact place with the sentence.

These values are ready-made features for any model.

TF-IDF Formula:

$$\text{TF (t, d)} = \text{count of t in doc / number of words in doc} \quad (8)$$

$$\text{IDF (t)} = \log(\text{N}/(\text{df} + 1)) \quad (9)$$

$$\text{TF-IDF (t, d)} = \text{TF (t, d)} * \log(\text{N}/(\text{df} + 1)) \quad (10)$$

The performance of TF-IDF in Arabic language is supports Arabic with full efficiency.

Similarity with Title Feature

This feature has been firstly proposed by Edmundson (Edmundson 1969) and defined as the similarity or the overlap between a given sentence and the document title. The importance of this feature comes from the idea that if a sentence consists of words appearing in the title, then it might be an important sentence. Moreover, if a sentence shares a key-phrase with the title, this will significantly increase its score. Therefore, the title similarity score for a sentence is computed using the following equation:

$$Title\ Similarity\ Score = Similarity\ (Title, Si) * \sqrt{1 + (KPT\ \cap\ KPS_i} \quad (11)$$

here Si is the current Sentence, KPT is the list of Key-Phrases that appear in document's title, KPSi is the list of Key-Phrases extracted from Sentence Si; (KPT ∩ KPSi) is the number of common key phrases between Si and T. The aim of using square root is to smoothly increase the score if the intersection is realized in more than one. Finally, Sim (Title; Si) is the degree of similarity between Si and the document's title computed by a cosine similarity measure, which is a well-known text similarity method (Gomaa & Fahmy 2013) (Shareghi & Hassanabadi 2008).

To compute the similarity, the sentence and the title are represented using the bag-of-words model. In this model, each sentence Si is represented as an N-dimensional vector Si = {wi1, wi2, …, wik, …win}, where wik is the weight of term tk that exists in the sentence Si, and n is the number of all possible unique words in the target document. Therefore, based on this representation, the cosine similarity can be computed as follows:

$$Cosine\ Similarity\ (Si,T) = \frac{\sum_{W\in S,T} \quad tf_{ws} * \ tf_{W,T}(isf_w)\hat{}2}{\sqrt{\sum_{S_i \epsilon S} \quad (tfS_iS * isfS_i)^2}\sqrt{\sum_{T_i \epsilon T} \quad (tfT_iT * isfT_i)^2}} \tag{12}$$

here tfw, Si is the frequency of word w in sentence S, which is defined as tfw, Si = 1 + log (tfw, Si), and isfw is the Inverse Sentence Frequency which is a special version of Inverse Document Frequency (IDF) that measures how much information a term provides. Thus, the term is considered important if it is dense in the given sentence and rare in the entire document (Doko, Stula, & Stipanicev, 2013). Inverse sentence frequency is defined as

$$isfw = log\ log\ \frac{N}{1 + \ |S_i \in \ S{:}W \in S_i|'} \tag{13}$$

where N is the number of sentences in the document and $|S_i \in S{:}W \in S_i|$ the number of sentences where the word w appears (Kirmani, Hakak, Mohd, & Mohd, 2019). For machine learning method, title feature is formulated depending on two features: the first one is computed using cosine similarity measure, as defined and the second one is represented as a binary value indicating the possibility for the sentence to share key-phrases with the title or not.

Algorithm 2. Proposed approach algorithm

1. Input: Entire Single Document.
2. Output: Output Document.
3. Configure/Set the Maximum Sentences in the Summary Total Sentences in Document.
4. Morphological Analyzer
5. Graph New Graph ()
6. Foreach Sentence: Document. Sentences
7. Normalization ()
8. Tokenization ()
9. StopWordsRemoval ()
10. Stemming ()
11. S_TF-IDF Calculate Sentence TF-IDF ()
12. S_Noun_List Applying Morphological Analyzer & Get Nouns List ()
13. New Node CreateGraphNode (S_TF-IDF, S_Noun_List)
14. Graph. Add (New Node)
15. Foreach Node: Graph. Nodes
16. If (Node <> New Node)
17. Cosin_Similarity (Node, New Node)
18. Nouns Measure Noun_Calc (Node)
19. Graph.CreateEdge(Node, New Node, Cosin_Similarity)
20. Foreach_sentence_set_the_number_of_its_nouns_as_initial_rank ()
21. Apply_Page_Rank ()

22. Summary Extract_Summary (Compression_Ratio)
23. Summary Removing_Reduandancy (Summary)
24. OUTPUT Summary

Summary Extraction

The summary is extracted on the basis of Compression Ratio (CR), which is the number of sentences that the summary consists of. This ratio represents a particular proportion of the number of sentences composes the original text that the user has a choice to choose it. After the construction of the graph and the calculation of edge weight, a weighted directed acyclic graph is established, but this graph could be irrelevant due to the fact that the original text could contain subtopics and sentences which do not contain any noun words, noun stems, or n-grams associated with other sentences. Consequently, no edges would link these sub graphs to each other, so the connection between the nodes should be verified before starting summary extraction. This phase consists of two steps; the verification of connection between graphs, and summary extraction. To verify that the graph is connected, we should comply with the following conditions:

1. Each node of a sentence should be related to at least one node of a sentence before and one node of a sentence after it, except the first and the last sentences.
2. The first node of a sentence should be related to at least one sentence after it. The last node of a sentence should be related to at least one sentence before it.

The aim of this step: to formulate a brief and sufficient content, so that it is sufficiently informative for the user, without using too many sentences.

Goal: produce an abridged version of a text that contains information that is important or relevant to a user.

Summarization Applications

1. Outlines or abstract of any documents, article, etc.
2. Summaries of email threads
3. Action items from a meeting
4. Simplifying text by compressing sentences

It has many applications such as:

Single-document summarization

1. Given a single document, produce.
    1.1. Abstract
    1.2. Outline
    1.3. Headline
2. Multiple-document summarization
3. Given a group of documents, produce a gist of content:
    3.1. A series of news stories on the same event
    3.2. A set of web pages about some topic or question

There are two types of summarizations.

The summary is from one file, through which we deal with one file, and conclude the summary and the introduction.

Summary from many files, in which that deal with several files on this point, and then produce a series of news or pages for one of the sites.

1. Generic summarization: summarize the content of a document.
2. Query-focused summarization:
   2.1. Summarize a document with respect to an information need expressed in a user query.
   2.2. A kind of complex question answering: Answer a question by summarizing a document that has the information to construct the answer.

Also, there are two other types of summarizations, based on the content itself.

1. General summary: which summarizes the content in general.
2. The customized summary, which summarizes the content depending on a specific research point. For example, the Battle of Stalingrad can be summarized from a specific aspect, which is the effect of the severe weather on both sides.

There are two other types of summarizations, depending on how:

1. Extractive summary: It selects words and sentences from the original text to be used as a summary of the content.
2. Abstractive structure: It is the one that formulates new words that may not be used in the original content. It is more efficient, but it is much more difficult.

This consists of three steps:

1. Content selection: choose sentences to extract from the document.
2. Information ordering: choose an order to place them in the summary.
3. Sentence realization: clean up the sentences.

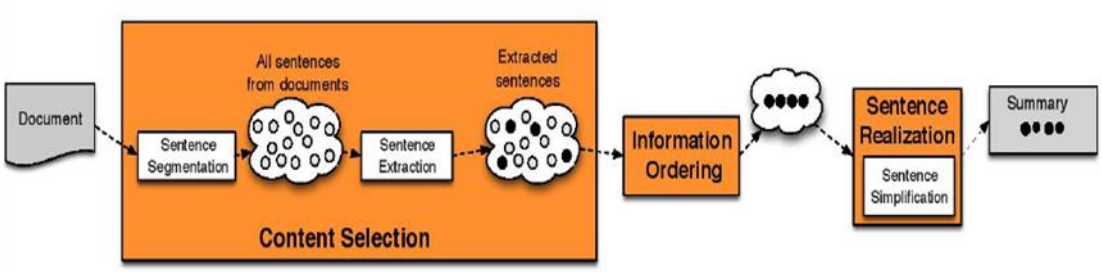

FIGURE 5. Flow chart of text summarization process

This is done in two ways, with or without supervision.

Unsupervised method, based on the idea of searching for informative or salient words.

Intuition dating back to Luhn (1958): Choose Sentences that have Salient or informative words.

Two approaches to defining salient words.

Tf-idf: weight each word Wi in document j by tf-idf

$$weight(W_i) = tf_{ij} \times idf_i \quad (14)$$

Topic signature: Choose a smaller set of salient words.

Mutual information

Log-likelihood ratio (LLR)

$$weight(W_i) = \{1 if -2\ log\ log\ \lambda\ (w_i) > 10\ \ 0 \qquad otherwise \quad (15)$$

This is done by calculating the weight using tf-idf or other equations, such as the word-to-word LLR equation 'if its value is more than 10.

Choose words that are informative either by log-likelihood ratio (LLR) or by appearing in the query.

$$weight(w_i) = \{1\ 1\ 0 \quad if -2\ log\ log\ \lambda(w_i) > 10\ \ if\ w_i \in question\ otherwise \quad (16)$$

weight a sentence (or window) by weight of its words:

$$weight(s) = \frac{1}{|S|} \sum_{w \in S} weight(w) \quad (17)$$

The weight is also calculated by making an equation so that it has a value of 1 if the LLR value is greater than 10, or if the word itself is present in the question, otherwise it is zero. Then the weight of the sentence is calculated by finding the average weights of the words in it, so that the sentence contains more words with a higher weight that has a preference.

This is done in two ways, with or without supervision:

1. Unsupervised method, based on the idea of searching for informative or salient words.
2. This is done by calculating the weight using tf-idf or other equations, such as the word-to-word LLR equation 'if its value is more than 10.
3. The weight is also calculated by making an equation so that it has a value of 1 if the LLR value is greater than 10, or if the word itself is present in the question, otherwise it is zero.
4. Then the weight of the sentence itself is calculated by finding the average weights of the words in it, so that the sentence contains more words with a higher weight that has a preference.

These are the steps of supervised Learning.

Given: A labeled training set of good summaries for each document

Align: The sentences in the document with sentences in the summary

Extract features: Position (first sentence?), Length of sentence, Word informativeness, cue phrases, Cohesion

Train: a binary classifier (put sentence in summary? Yes or no)

Problems: hard to get labeled training data, alignment difficult, performance not better than unsupervised algorithms

So, in practice: Unsupervised content selection is more common.

So how is this summarization evaluated?

Summarization is evaluated through the ROUGE method, which is the abbreviation of the complete evaluation of the essence of the meaning.

Its idea is based on the following steps:

1. A number of individuals will be summarizing the required paragraph, for a specific number of words, let it be 10 words.
2. The algorithm summarizes the paragraph itself.
3. now will be compared between summarizing algorithms and summarizing individuals in a specific way.
4. to calculate the number of bigrams shared between summarizing the algorithm and summarizing the first individual, and adding them to the number of bigrams shared with the second individual, and so on...
5. divided the total by the total number of bigrams for all individuals.

ROUGE (Recall Oriented Understudy for Gisting Evaluation)
Intrinsic metric for automatically evaluating summaries.

1. Based on BLEU (a metric used for machine translation)
2. Not as good as human evaluation (Did this answer the user's question?)
3. But much more convenient

Given a document D, and an automatic summary X:

1. Have N humans produce a set of reference summaries of D
2. Run system, giving automatic summary X
3. What percentage of the bigrams from the reference summaries appear in X?

$$ROUGE-2 = \frac{\sum_{S\in\{RefSummaries\}} \sum_{bigrams\ i\in S} min(count(i,x), count(i,S))}{\sum_{S\in\{RefSummaries\}} \sum_{bigrams\ i\in S} count(i,S)} \quad (18)$$

A higher number of NGrams can be used, but the efficiency will be inexpressive and imprecise. Unigrams cannot be used.

A ROUGE example: Q:" What is water spinach?"

Human1: water spinach is a green leafy vegetable grow in the tropics.

Human2: water spinach is a semi-aquatic tropical plant grown as a vegetable.

Human3: water spinach is a commonly eaten leaf vegetable of Asia.

System answer: water spinach is a leaf vegetable commonly eaten in tropical areas of Asia.

$$ROUGE - 2 = \frac{3+3+6}{10+9+9} = \frac{12}{28} = 0.43 \quad (19)$$

If we had a summary of 3 individuals, and a summary of the algorithm, we see that the summary of the algorithm shared with the summary of the first individual in 3 bigrams, which are: water spinach, spinach is, is a
The second is the same, and the third participates with it in 6, so the total is 12, and we divide it by the total number of bigrams for the three sentences, which is 28, so the efficiency is 0.43, and the more the summarization of the algorithm participates with the summarization of individuals, the greater the efficiency.

Feature Extraction and Formulation
An extractive-based text summary that involves selecting sentences of high relevance or importance is based on employing a set of features to generate coherent summaries that state the main idea of the given document. Therefore, selecting and designing these features will greatly affect the quality of the generated summaries.

These features are classified into four levels including word-based level, sentence-based level, paragraph-based level, and graph-based features. Since the quality of the generated extractive summary is highly affected by the selected features along with their design, our target in this Research is to redesign the most important or prominent features that identify the most important sentences in addition to maximizing content coverage and diversity between sentences within the summary.

Using statistical features alone may not provide rich information summary, because they don't take into consideration the meaning and may cause some redundancy in the generated summary.

The performance in the Arabic language Extractive summary is good.

The performance in the Arabic language Abstractive summary is not good and not accurate.

EXPERIMENTATION AND RESULTS

Data Set (Corpus)
To evaluate the proposed approach, the Essex Arabic Summaries Corpus (EASC) is used as a standard corpus, the corpus contains 153 documents, and each document has 5 summaries, with a total of 765 Arabic human-made summaries generated using Mechanical (El-Haj et al. 2011). EASC includes 10 subjects: art, music, environment, politics, sports, health, finance, science and technology, tourism, religion, and education.

ROUGE-N AND BLEU METRICES

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is a set of metrics that is used to score a machine-generated summary using one or more reference summaries created by humans. ROUGE-N is the evaluation of N-grams recall over all the reference summaries. The recall is calculated by dividing the number of overlapping words over the total number of words in the reference summary (Lin 2004). The BLEU metric, contrary to ROUGE, is based on N-grams precision. It refers to the percentage of the words in the machine generated summary overlapping with the reference summaries (Papineni et al. 2002). ROUGE-N is considered the most popular one. It counts the number of overlapping units between the computer-generated summary and the reference summaries which can be computed using the following formula:

$$ROUGE - N = \frac{\sum_{S \in Summ_{ref}} \sum_{N-grams \in S} count_{match}(N-gram)}{\sum_{S \in Summ_{ref}} \sum_{N-grams \in S} count(N-gram)} \quad (20)$$

where V is the length of the N-gram, Count match (N - gram) is the maximum number of the common N-grams between the set of reference summaries (Summref) and the generated summary, and Count (N - grams) is the total number of n-grams in the reference summary. There are many variations of ROUGE-N depending on the unit size. The most used ones which are used by DEC 2007 are ROUGE-1 and ROUGE-2. Since ROUGE-N is a recall-oriented measure, Precision P, Recall R, and F-score can be defined as follows:

$$P = \frac{\left|grams_{ref} \cap grams_{gen}\right|}{grams_{gen}}, R = \frac{\left|grams_{ref} \cap grams_{gen}\right|}{grams_{ref}}, F1 = \frac{2PR}{P+R} \quad (21)$$

Experiments Setup and Results

The goal of the proposed experiment is to achieve the following results:

1. evaluating the proposed design of the selected statistical and semantic features,
2. evaluating the application of a statistical summarization method on the Arabic texts and,
3. comparing our proposed method to other related works.

As mentioned earlier, the EASC dataset has been used in experimenting and evaluating the proposed method. For evaluation measures, ROUGE-N (e.g. ROUGE-1, ROUGE-2) is used where the precision, recall, and F-score are calculated for each of the generated summaries for both summary methods.

TABLE 7. Performance of score-based summarization method

Reference Summary	ROUGE-1			ROUGE-2		
	Recall	Precision	F-score	Recall	Precision	F-score
Five reference summaries	0.513	0.388	0.442	0.382	0.313	0.344
Gold-Standard reference summary	0.673	0.616	0.643	0.633	0.601	0.617

CONCLUSIONS

Text summarization is considered very useful for readers to understand the main idea of provided long text and saves time and effort. Although there are many studies related to text summarization of different languages, Arabic text summarization is considered a greenfield for exploring and doing new research. This study shows the best algorithm used to provide higher accuracy and quality for Arabic text summarization for single and multi-values summarization. The goal of this research is to study the interaction between a set of statistical and semantic features and their impact on the process of extractive text summarization with the final objective of selecting the most significant features. The obtained results have shown that semantic-based methods stop word removal, lemmatization, POS tagging and word sense disambiguation improves the resultant summary. Redundant information is also detected to produce more accurate results. For the evaluation of the results, ROUGE scores are used. Comparison of the results against other text summarization approaches is also done. We introduced TextRank – a graph based ranking model for text how it can be successfully used for natural language applications. In particular, we proposed and evaluated two innovative unsupervised approaches for keyword and sentence extraction, and showed that the accuracy achieved by TextRank in these applications is competitive with that of previously proposed state-of-the-art algorithms.

REFERENCES

Abbasi-ghalehtaki, R., Khotanlou, H., & Esmaeilpour, M. (2016). Fuzzy evolutionary cellular learning automata model for text summarization. *Swarm and Evolutionary Computation, 30*, 11-26.

Al-Abdallah, R. Z., & Al-Taani, A. T. (2017). Arabic single-document text summarization using particle swarm optimization algorithm. *Procedia Computer Science, 117*, 30-37.

AL-Khawaldeh, F. T., & Samawi, V. W. (2015). Lexical cohesion and entailment based segmentation for arabic text summarization (lceas). *World of Computer Science & Information Technology Journal, 5*(3).

Al-Radaideh, Q. A., & Bataineh, D. Q. (2018). A hybrid approach for arabic text summarization using domain knowledge and genetic algorithms. *Cognitive Computation, 10*(4), 651-669.

Al-Taani, A. T., & Al-Omour, M. M. (2014). *An extractive graph-based Arabic text summarization approach.* Paper presented at the The International Arab Conference on Information Technology.

Althobaiti, M., Kruschwitz, U., & Poesio, M. (2014). AraNLP: A Java-based library for the processing of Arabic text.

Amato, F., Castiglione, A., Mercorio, F., Mezzanzanica, M., Moscato, V., Picariello, A., & Sperlì, G. (2018). Multimedia story creation on social networks. *Future Generation Computer Systems, 86*, 412-420.

Amato, F., Marrone, S., Moscato, V., Piantadosi, G., Picariello, A., & Sansone, C. (2019). HOLMeS: EHealth in the big data and deep learning era. *Information, 10*(2), 34.

Aries, A., Zegour, D. E., & Hidouci, K. W. (2015). *AllSummarizer system at MultiLing 2015: Multilingual single and multi-document summarization.* Paper presented at the Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue.

Asher, N., Asher, N. M., & Lascarides, A. (2003). *Logics of conversation*: Cambridge University Press.

Ayedh, A., Tan, G., Alwesabi, K., & Rajeh, H. (2016). The effect of preprocessing on arabic document categorization. *Algorithms, 9*(2), 27.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Belkebir, R., & Guessoum, A. (2015). A supervised approach to arabic text summarization using adaboost. In *New contributions in information systems and technologies* (pp. 227-236): Springer.

Chowdhury, G. G. (2003). Natural language processing. *Annual review of information science and technology, 37*(1), 51-89.

Dalal, V., & Malik, L. (2013). *A survey of extractive and abstractive text summarization techniques.* Paper presented at the 2013 6th International Conference on Emerging Trends in Engineering and Technology.

Doko, A., Stula, M., & Stipanicev, D. (2013). A recursive TF-ISF based sentence retrieval method with local context. *International Journal of Machine Learning and Computing, 3*(2), 195.

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the ACM (JACM), 16*(2), 264-285.

El-Haj, M., Kruschwitz, U., & Fox, C. (2011). *Exploring clustering for multi-document Arabic summarisation.* Paper presented at the Asia Information Retrieval Symposium.

El-Khair, I. A. (2006). Effects of stop words elimination for Arabic information retrieval: a comparative study. *International Journal of Computing & Information Sciences, 4*(3), 119-133.

Fattah, M. A., & Ren, F. (2009). GA, MR, FFNN, PNN and GMM based models for automatic text summarization. *Computer Speech & Language, 23*(1), 126-144.

Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications, 68*(13), 13-18.

Gupta, V., & Lehal, G. S. (2010). A survey of text summarization extractive techniques. *Journal of emerging technologies in web intelligence, 2*(3), 258-268.

Hasan, K. S., & Ng, V. (2010). *Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art.* Paper presented at the Coling 2010: Posters.

Imam, I., Nounou, N., Hamouda, A., & Khalek, H. A. A. (2013). An ontology-based summarization system for arabic documents (ossad). *International Journal of Computer Applications, 74*(17), 38-43.

Kanapala, A., Pal, S., & Pamula, R. (2019). Text summarization from legal documents: a survey. *Artificial Intelligence Review, 51*(3), 371-402.

Khan, A., & Salim, N. (2014). A review on abstractive summarization methods. *Journal of Theoretical and Applied Information Technology, 59*(1), 64-72.

Kirmani, M., Hakak, N. M., Mohd, M., & Mohd, M. (2019). Hybrid text summarization: a survey. In *Soft Computing: Theories and Applications* (pp. 63-73): Springer.

Lagrini, S., Redjimi, M., & Azizi, N. (2017). Automatic arabic text summarization approaches. *International Journal of Computer Applications, 164*(5), 31-37.

Lin, C.-Y. (2004). *Rouge: A package for automatic evaluation of summaries.* Paper presented at the Text summarization branches out.

Lloret, E., Ferrández, O., Munoz, R., & Palomar, M. (2008). *A Text Summarization Approach under the Influence of Textual Entailment.* Paper presented at the NLPCS.

Mallick, C., Das, A. K., Dutta, M., Das, A. K., & Sarkar, A. (2019). Graph-based text summarization using modified TextRank. In *Soft computing in data analytics* (pp. 137-146): Springer.

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text, 8*(3), 243-281.

Mathkour, H. (2014). A Novel Rhetorical Structure Approach for Classifying Arabic Security Documents. *IJCI. International Journal of Computers and Information, 3*(1), 15-27.

Mihalcea, R., & Tarau, P. (2004). *Textrank: Bringing order into text.* Paper presented at the Proceedings of the 2004 conference on empirical methods in natural language processing.

Mustafa, M., Eldeen, A. S., Bani-Ahmad, S., & Elfaki, A. O. (2017). A comparative survey on arabic stemming: approaches and challenges. *Intelligent Information Management, 9*(02), 39.

Nallapati, R., Zhou, B., & dos Santos, C. (2016). *glar Gulçehre, C.; and Xiang, B. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond.* Paper presented at the Proceedings of CoNLL.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). *Bleu: a method for automatic evaluation of machine translation.* Paper presented at the Proceedings of the 40th annual meeting of the Association for Computational Linguistics.

Ramos, J. (2003). *Using tf-idf to determine word relevance in document queries.* Paper presented at the Proceedings of the first instructional conference on machine learning.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., & Perez, C. J. (2019). Comparison of automatic methods for reducing the Pareto front to a single solution applied to multi-document text summarization. *Knowledge-Based Systems, 174*, 123-136.

Shareghi, E., & Hassanabadi, L. S. (2008). *Text summarization with harmony search algorithm-based sentence extraction.* Paper presented at the Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology.

Sobh, I., Darwish, N., & Fayek, M. (2009). An optimized dual classification system for Arabic extractive generic text summarization. *Giza, Egypt*.

Suleiman, D., Awajan, A. A., & Al Etaiwi, W. (2019). *Arabic text keywords extraction using Word2vec.* Paper presented at the 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS).

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems, 27*.

Tatar, D., Mihis, A., Lupsa, D., & Tamaianu-Morita, E. (2009). Entailment-based linear segmentation in summarization. *International Journal of Software Engineering and Knowledge Engineering, 19*(08), 1023-1038.

Verma, P., & Om, H. (2019). MCRMR: Maximum coverage and relevancy with minimal redundancy based multi-document summarization. *Expert systems with applications, 120*, 43-56.

Waheeb, S. A., Khan, N. A., Chen, B., & Shang, X. (2020). Multidocument arabic text summarization based on clustering and Word2Vec to reduce redundancy. *Information, 11*(2), 59.

Wan, X., & Yang, J. (2008). *Multi-document summarization using cluster-based link analysis.* Paper presented at the Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval.

Wei, F., Li, W., Lu, Q., & He, Y. (2010). A document-sensitive graph model for multi-document summarization. *Knowledge and information systems, 22*(2), 245-259.

Zou, W. Y., Socher, R., Cer, D., & Manning, C. D. (2013). *Bilingual word embeddings for phrase-based machine translation.* Paper presented at the Proceedings of the 2013 conference on empirical methods in natural language processing.

APPENDIX A: TFI-DF (Term Frequency-Inverse Document Frequency)

```
import pandas as pd
documentA = 'ذهب محمد الي الجامعة ليدرس الفيزياء و الكيمياء'
documentB = 'ذاكرت مني الرياضيات و الفيزياء في الجامعة'
bagOfWordsA = documentA.split(' ')
bagOfWordsB = documentB.split(' ')
uniqueWords = set(bagOfWordsA).union(set(bagOfWordsB))
uniqueWords
{'الجامعة', 'الرياضيات', 'الفيزياء', 'الكيمياء', 'الي', 'ذاكرت', 'ذهب', 'في', 'ليدرس', 'محمد', 'مني', 'و'}
numOfWordsA = dict.fromkeys(uniqueWords, 0)
for word in bagOfWordsA:
numOfWordsA[word] += 1
numOfWordsA
{'الكيمياء': 1,
 'ذهب': 1,
 'ذاكرت': 0,
 'و': 1,
 'الي': 1,
 'ليدرس': 1,
 'الفيزياء': 1,
 'الرياضيات': 0,
 'مني': 0,
 'محمد': 1,
 'الجامعة': 1,
 'في': 0}
numOfWordsB = dict.fromkeys(uniqueWords, 0)
```

```
for word in bagOfWordsB:
numOfWordsB[word] += 1
numOfWordsB
{'الكيمياء': 0,
'ذهب': 0,
 'ذاكرت': 1,
 'و': 1,
 'الي': 0,
 'ليدرس': 0,
 'الفيزياء': 1,
 'الرياضيات': 1,
 'مني': 1,
 'محمد': 0,
 'الجامعة': 1,
 'في': 1}
def computeTF(wordDict, bagOfWords):
tfDict = {}
bagOfWordsCount = len(bagOfWords)
for word, count in wordDict.items():
tfDict[word] = count / float(bagOfWordsCount)
return tfDict
tfA = computeTF(numOfWordsA, bagOfWordsA)
tfA
{'الكيمياء': 0.125,
 'ذهب': 0.125,
 'ذاكرت': 0.0,
 'و': 0.125,
```

```
 'الي': 0.125,
 'ليدرس': 0.125,
 'الفيزياء': 0.125,
 'الرياضيات': 0.0,
 'مني': 0.0,
 'محمد': 0.125,
 'الجامعة': 0.125,
 'في': 0.0}
tfB = computeTF(numOfWordsB, bagOfWordsB)
tfB
{'الكيمياء': 0.0,
 'ذهب': 0.0,
 'ذاكرت': 0.14285714285714285,
 'و': 0.14285714285714285,
 'الي': 0.0,
 'ليدرس': 0.0,
 'الفيزياء': 0.14285714285714285,
 'الرياضيات': 0.14285714285714285,
 'مني': 0.14285714285714285,
 'محمد': 0.0,
 'الجامعة': 0.14285714285714285,
 'في': 0.14285714285714285}
def computeIDF(documents):
import math
N = len(documents)
idfDict = dict.fromkeys(documents[0].keys(), 0)
for document in documents:
```

```
for word, val in document.items():
if val > 0:
idfDict[word] += 1
for word, val in idfDict.items():
idfDict[word] = math.log(N / float(val))
return idfDict
idfs = computeIDF([numOfWordsA, numOfWordsB])
idfs
{'الكيمياء': 0.6931471805599453,
 'ذهب': 0.6931471805599453,
 'ذاكرت': 0.6931471805599453,
 'و': 0.0,
 'الي': 0.6931471805599453,
 'ليدرس': 0.6931471805599453,
 'الفيزياء': 0.0,
 'الرياضيات': 0.6931471805599453,
 'مني': 0.6931471805599453,
 'محمد': 0.6931471805599453,
 'الجامعة': 0.0,
 'في': 0.6931471805599453}
def computeTFIDF(tfBagOfWords, idfs):
tfidf = {}
for word, val in tfBagOfWords.items():
tfidf[word] = val * idfs[word]
return tfidf
tfidfA = computeTFIDF(tfA, idfs)
tfidfA
```

{'الكيمياء': 0.08664339756999316,

'ذهب': 0.08664339756999316,
'ذاكرت ': 0.0,

'و': 0.0,

'الي': 0.08664339756999316,

'ليدرس': 0.08664339756999316,

'الفيزياء': 0.0,

'الرياضيات': 0.0,

'مني': 0.0,

'محمد': 0.08664339756999316,

'الجامعة': 0.0,

'في': 0.0}

```
tfidfB = computeTFIDF(tfB, idfs)
tfidfB
```

{'الكيمياء': 0.0,

'ذهب': 0.0,

'ذاكرت': 0.09902102579427789,

'و': 0.0,

'الي': 0.0,

'ليدرس': 0.0,

'الفيزياء': 0.0,

'الرياضيات': 0.09902102579427789,

'مني': 0.09902102579427789,

'محمد': 0.0,

'الجامعة': 0.0,

'في': 0.09902102579427789}

```
df = pd.DataFrame([tfidfA, tfidfB])
```

```
df.head()
```

	الكيمياء	ذهب	ذاكرت	و	الي	ليدرس	الفيزياء	الرياضيات	متي	محمد	الجامعة	في
0	0.086643	0.086643	0.000000	0.0	0.086643	0.086643	0.0	0.000000	0.000000	0.086643	0.0	0.000000
1	0.000000	0.000000	0.099021	0.0	0.000000	0.000000	0.0	0.099021	0.099021	0.000000	0.0	0.099021

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()

vectors = vectorizer.fit_transform([documentA, documentB])

print(vectors)
```

```
(0, 3)        0.4078241041497786
(0, 2)        0.29017020899133733
(0, 8)        0.4078241041497786
(0, 0)        0.29017020899133733
(0, 4)        0.4078241041497786
(0, 9)        0.4078241041497786
(0, 6)        0.4078241041497786
(1, 7)        0.4466561618018052
(1, 1)        0.4466561618018052
(1, 10)       0.4466561618018052
(1, 5)        0.4466561618018052
(1, 2)        0.31779953783628945
(1, 0)        0.31779953783628945
```

```
feature_names = vectorizer.get_feature_names()

feature_names
```

```
['الجامعة',
'الرياضيات',
 'الفيزياء',
 'الكيمياء',
```

```
'الي',
'ذاكرت',
'ذهب',
'في',
'ليدرس',
'محمد',
'مني']
dense = vectors.todense()
dense
matrix([[0.29017021, 0.        , 0.29017021, 0.4078241 , 0.4078241 ,
0.        , 0.4078241 , 0.        , 0.4078241 , 0.4078241 ,
0.        ],
[0.31779954, 0.44665616, 0.31779954, 0.        , 0.        ,
0.44665616, 0.        , 0.44665616, 0.        , 0.        ,
0.44665616]])
denselist = dense.tolist()
df = pd.DataFrame(denselist, columns=feature_names)
df.head()
```

	الجامعة	الرياضيات	الفيزياء	الكيمياء	الي	ذاكرت	ذهب	في	ليدرس	محمد	مني
0	0.29017	0.000000	0.29017	0.407824	0.407824	0.000000	0.407824	0.000000	0.407824	0.407824	0.000000
1	0.31780	0.446656	0.31780	0.000000	0.000000	0.446656	0.000000	0.446656	0.000000	0.000000	0.446656

APPENDIX B: Arabic Text Summarization

```
pip install nltk
pip install Arabic-Stopwords
# importing libraries
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
nltk.download('stopwords')
nltk.download('punkt')
stopwords = stopwords.words('arabic')
import bs4 as bs
import urllib.request
import re

# Input text - to summarize

text = """
عبد الله محمد بن موسى الخوارزمى عالم رياضيات وفلك وجغرافيا مسلم. يكنى باسم الخوارزمى وأبي جعفر. قيل أنه ولد حوالي 164هـ 781م (وهو غير مؤكد) وقيل أنه توفي بعد 232 هـ أي (بعد 847م). يعتبر من أوائل علماء الرياضيات الم سلمين حيث ساهمت أعماله بدور كبير فى تقدم الرياضيات فى عصره. اتصل بالخليفة العباسى المأمون وعمل فى بيت ال حكمة فى بغداد وكسب ثقة الخليفة إذ ولاه المأمون بيت الحكمة كما عهد إليه برسم خارطة للأرض عمل فيها أكثر من سبعين جغرافيا. قبل وفاته فى 850 م/232 هـ كان الخوارزمى قد ترك العديد من المؤلفات فى علوم الرياضيات والفلك وا لجغرافيا ومن أهمها كتاب المختصر في حساب الجبر والمقابلة الذي يعد أهم كتبه.

ترجم الكتابَ إلى اللغة اللاتينية حوالي عام 1145 م العالم روبرت مِن تشستر. دخلت على إثر ذلك كلمات مثل الجبر Al gebra والصفر Zero إلى اللغات اللاتينية وترجمه بعد ذلك بقليل جيراردو الكريموني الساكن في طليطلة، متبوعا في ذل ك بترجمة ثالثة قام بها الإيطالي غيوم دي لونا. استُعملت ترجمة روبرت مِن تشستر الكتاب الرئيسى في الرياضيات إلى حد ود القرن السادس عشر في الجامعات الأوروبية.
"""

# Tokenizing the text
stopWords = set(stopwords.words("arabic"))
words = word_tokenize(text)

# Creating a frequency table to keep the
# score of each word

freqTable = dict()
for word in words:
  word = word.lower()
  if word in stopWords:
    continue
  if word in freqTable:
```

```
    freqTable[word] += 1
  else:
    freqTable[word] = 1

# Creating a dictionary to keep the score
# of each sentence
sentences = sent_tokenize(text)
sentenceValue = dict()

for sentence in sentences:
  for word, freq in freqTable.items():
    if word in sentence.lower():
      if sentence in sentenceValue:
        sentenceValue[sentence] += freq
      else:
        sentenceValue[sentence] = freq

sumValues = 0
for sentence in sentenceValue:
  sumValues += sentenceValue[sentence]

# Average value of a sentence from the original text
average = int(sumValues / (len(sentenceValue)+1))

# Storing sentences into our summary.
summary = ''
for sentence in sentences:
  if (sentence in sentenceValue) and (sentenceValue[sentence] > (1.2 * average)):
    summary += " " + sentence

print(summary)
```

اتصل بالخليفة العباسي المأمون وعمل في بيت الحكمة في بغداد وكسب ثقة الخليفة إذ ولاه المأمون بيت الحكمة كما عهد إليه برسم خارطة للأرض عمل فيها أكثر من سبعين جغرافيا. قبل وفاته في 850 م/232 هـ كان الخوارزمي قد ترك العديد من المؤلفات في علوم الرياضيات والفلك والجغرافيا ومن أهمها كتاب المختصر في حساب الجبر والمقابلة الذي يعد أهم كتبه. دخلت على إثر ذلك كلمات مثل الجبر Algebra والصفر Zero إلى اللغات اللاتينية وترجمه بعد ذلك بقليل جيراردو الكريموني الساكن فى طليطلة، متبوعا فى ذلك بترجمة ثالثة قام بها الإيطالى غيوم دي لونا. استُعملت ترجمة روبرت مِن تشستر الكتاب الرئيسي في الرياضيات إلى حدود القرن السادس عشر في الجامعات الأوروبية.