electronics

MDPI

Article

Extractive Arabic Text Summarization-Graph-Based Approach

Yazan Alaya AL-Khassawneh [1,*] and Essam Said Hanandeh [2]

[1] Data Science and Artificial Intelligence Department, Zarqa University, Zarqa P.O. Box 13110, Jordan
[2] Computer Information Systems Department, Zarqa University, Zarqa P.O. Box 13110, Jordan
* Correspondence: ykhassawneh@zu.edu.jo

Abstract: With the noteworthy expansion of textual data sources in recent years, easy, quick, and precise text processing has become a challenge for key qualifiers. Automatic text summarization is the process of squeezing text documents into shorter summaries to facilitate verification of their basic contents, which must be completed without losing vital information and features. The most difficult information retrieval task is text summarization, particularly for Arabic. In this research, we offer an automatic, general, and extractive Arabic single document summarizing approach with the goal of delivering a sufficiently informative summary. The proposed model is based on a textual graph to generate a coherent summary. Firstly, the original text is converted to a textual graph using a novel formulation that takes into account sentence relevance, coverage, and diversity to evaluate each sentence using a mix of statistical and semantic criteria. Next, a sub-graph is built to reduce the size of the original text. Finally, unwanted and less weighted phrases are removed from the summarized sentences to generate a final summary. We used Recall-Oriented Research to Evaluate Main Idea (RED) as an evaluative metric to review our proposed technique and compare it with the most advanced methods. Finally, a trial on the Essex Arabic Summary Corpus (EASC) using the ROUGE index showed promising results compared with the currently available methods.

Keywords: extractive Arabic text summarization; graph-based summarization; feature extraction; triangle counting

Citation: AL-Khassawneh, Y.A.; Hanandeh, E.S. Extractive Arabic Text Summarization-Graph-Based Approach. *Electronics* **2023**, *12*, 437. https://doi.org/10.3390/electronics12020437

Academic Editor: Dah-Jye Lee

Received: 2 October 2022
Revised: 3 December 2022
Accepted: 11 December 2022
Published: 14 January 2023

1. Introduction

The huge amount of digital text data produced each day makes it more and more difficult to quickly and accurately retrieve important information from texts [1]. To obtain this data, an Automated Text Summarization (ATS) can be created. In order to solve this issue and enable Arabic Natural Language Processing (NLP) systems, specialized Arabic ATS techniques are required. Computerized textual content summarization means using a gadget or primarily computer-based equipment to supply a useful precis. Although primary computerized textual content summarization solutions were introduced in the 1950s [2,3], summarization has been lengthy and is one of the important challenges of natural language processing. Because machines have a very difficult time grasping a text's substance based on its syntactic and semantic structure, computer-generated summaries frequently differ from those created by humans [4]. Systems for summarizing information can be categorized according to the type of input, output, goal, language, and summary technique. Summarization systems are classified into two categories based on the variety of input documents: single-document and multi-document. A summarization system's goals can vary depending on the type of input it receives, such as plain text, news articles, scientific articles, etc. These goals can include generating current information, running queries, or educating users about a particular topic. The method of summarizing is frequently heavily influenced by the goal for which it is being performed [5]. The two types of summarization techniques are extractive and abstractive. The process of extractive summarization entails choosing a group of sentences or phrases from the text depending on the scores they receive in accordance with a specified criterion and pasting them verbatim into the summary. A

concise interpretation of the original text is what is referred to as an abstractive summary. With this approach, the summary's sentences may not always be written exactly as they were in the original text. The purpose for which summarization systems are designed can also be used to categorize them as either educational or informative.

Most present automated summarization structures are used with the extractive summarization approach. Extractive summarization can be accomplished with three strategies: statistical approach, linguistic method, and mixed method [4]:

Statistical approach: This method of summary relies on the quantitative properties of the text and the statistical distribution of the features of interest. This method relies on information retrieval and classification methods without attempting to comprehend the entirety of the material. In this technique, an information retrieval algorithm examines the placement, length, and frequency of words and sentences in the document, and a classifier, using a collection of cases on which it has been trained, assesses which phrases could be included in the summary. With this approach, the original text's sentences are taken out without considering the words' semantics.

Linguistic approach: In this method, the computer must possess a thorough understanding of the language it is processing in order to analyze and comprehend sentences and select the phrases that should be included in the summary. This approach uses part-of-speech tagging, grammatical analysis, lexical analysis, and the extraction of significant phrases to determine links between words and phrases in the text. Sign words, characteristics, nouns, and verbs could all be used as the parameters for these processes. The linguistic technique frequently results in superior summaries because it takes into account the semantic relationships in the original text, even though the statistical approach is typically more computationally efficient.

Combination method: to produce more succinct and insightful summaries, this method combines both statistical and linguistic techniques. While statistical summarizing approaches are quite basic and adaptable because they use statistical features, they are also more prone to incoherence and inconsistently generated summaries.

The quality of output summaries can be greatly improved by combining several extractive summarization approaches. Based on the linguistic features extracted from the text structure analysis, modeling of the text structure and the relationships between its entities, and an improved single-document feature selection process, the combined approach to summarization is used in this study to produce unambiguous, succinct, consistent, and coherent summaries.

We suggested an extractive graph-based Arabic ATS technique in this research. It also describes how the choice of the phrase's fundamental component—the stem, word, or n-gram, which serves as the foundation for the calculations of similarity and sentence ranking (summarization processes)—can affect the efficacy of the extracted summary.

The graph-based Arabic ATS method is based on the method developed by [6]. Thakkar suggested a method for extracting the summary from a given English document by representing it as an undirected graph where sentences are represented by nodes, and the similarities (which refers to the word overlap) between every two sentences are represented by the edge weight. A summary is then generated by determining the shortest path between the first and remaining sentences of the original document. Moving from the first sentence to the last sentence broadens the summary and is more likely to include the most important parts of the original text.

2. Related Work

Luhn first proposed the idea of automatic text summarization in 1958, in the sense of figuring out how words are distributed inside sentences and identifying the document's keywords [7]. Since then, numerous summarizing techniques have been created using various methodologies and for various objectives. However, the majority of these approaches can be seen as advancements over earlier strategies. In this section, we concentrate on the

studies using graph-based extraction techniques for a single document. We also explore research that introduced Arabic text summarizing tools.

Recently, several graph-based approaches for summarizing single and multiple English documents have been developed. Among these strategies are [8–11].

The researchers in ref. [12] used a graph-based approach to extractive summarization. The later researchers suggested a brand-new summarizing technique based on a hybrid modeling graph. They suggested implementing a cutting-edge hybrid similarity function (H) that combines four different similarity measurements: cosine, Jaccard, word alignment, and window-based similarity. The method makes use of a trainable summarizer and takes into account a number of factors. It has been investigated how certain characteristics affect the work of summarization.

In ref. [13], a graph reduction technique known as the Triangle Counting Method is developed to select essential phrases in the text. The initial stage is to visualize a text as a graph, where the phrases serve as the nodes, and the similarities between them serve as the edges. Following the representation of the bit vector, the creation of triangles comes next, and acquiring phrases based on the bit vector's values comes last. This study demonstrated that it is possible to change one graph into another with a significantly smaller number of triangles. Adjacency Matrix Representation is simple to use and has sped up implementation times.

Two Arabic summarizing systems were created by El-Haj et al. [14,15]. The Arabic Question-Based Single Text Summarizer System (AQBTSS) works with an Arabic document and an Arabic query to provide a summary that is appropriate for the query of the document. The second system, called the Arabic Concept-Based Text Summarization System (ACBTSS), uses a set of words that reflect a certain concept as its input rather than a user's query. The first two phases of the two systems are the same: selecting a document from the document collection that matches the user's query and breaking the text up into sentences. Both systems use the Vector Space Model (VSM) in the summarization phase, where the weighting scheme is based on VSM and uses two measures, term frequency and inverse document frequency. In AQBTSS, each sentence is compared to the user query to find relevant sentences, whereas in ACBTSS, each sentence is matched against a set of keywords that represent a given concept. A panel of 1500 users evaluated the readability of the summaries of 251 articles produced by the two systems to evaluate them. The results revealed that AQBTSS performed better than ACBTSS.

A platform for summarizing Arabic texts was proposed by [16] and includes the following modules: tokenization, morphological analyzer, parser, relevant sentence extraction, and extract revision. A variety of texts (short, average, and long) were used in the evaluation of this platform in terms of execution time, and it was discovered that the run time of the platform's modules for a specific text was influenced by its size, i.e., the shorter the text, the weaker its run time.

The Sakhr Summarizer is an Arabic summarization tool that extracts the key phrases from the source text and summarizes them [17]. The Summarization engine makes use of the Sakhr Corrector to automatically correct the input Arabic text for frequent grammatical errors and the Keywords Extractor to find a prioritized list of keywords to accurately identify the essential phrases.

Authors in ref. [18] suggested a different summary system—the Arabic Intelligent Summarizer. The main machine-supervised learning technique is the foundation of this system. There are two phases to the system. The learning phase, which uses SVMs, is the first and instructs the algorithm on how to extract summary sentences. The users can summarize a new document during the use phase—the second stage.

P.M. Sabuna and D.B. Setyohadi [19] describe the development of an abstractive automatic summarization system for online discussion papers using the vector space concept. The three modules that make up this system are point curation, point extraction, and summary creation. By dependency parsing and examining the grammatical structure,

points are extracted. Shorter points are created by smaller indirect points after choosing the topic points and the points that might work for the summary.

An extractive summary technique for Arabic texts has been developed in [20]. This approach combines rhetorical structure theory (RST), one of the most popular theories in natural language processing, with semantic data taken from the Arabic word net. The quality of Arabic text summarization is improved using this method, which combines linguistic selection methods with sentence feature selection methods. In order to determine how closely related sentences are to the main title and subheadings, the suggested RST-based method first constructs an initial summary and then uses the score of each sentence in that summary.

The automatic Indonesian text summarizing system described in [21] generates summaries by combining sentence scores and decision trees. The C4.5 algorithm is employed in this system to pick the sentences that are of interest. After that, each sentence is scored using a sentence-scoring approach that takes into account eight variables, including TF-IDF, uppercase letters, proper nouns, cue phrases, numerical data, sentence length, sentence position, and title similarity. Following the creation of a decision tree model using the training data, the important sentences are identified, and the summary is prepared using the model's rules. A combined statistical-linguistic approach-based extractive summary technique for Indian literature has been described in [22]. Preprocessing, sentence feature extraction, and genetic algorithm (GA) for ranking sentences based on optimum feature weights are the three primary components of this summarization method. A sentence feature vector serves as a representation for each sentence. The statistical-linguistic properties of each sentence are analyzed, and a score is generated based on the importance of the features in that sentence. The sentences are then ranked based on the findings. Sentence characteristics accept values in the range of 0 to 1. After a predetermined number of generations in the GA, the fittest chromosome is chosen, and the Euclidean distance formula is used to calculate the distance between each sentence score and the fittest chromosome. The sentences are then arranged according to increasing distance. Finally, a summary is created by selecting a specific number of the document's top-ranked sentences, depending on the level of summarization that is desired.

Authors in ref. [23] suggested a multi-morphological analysis-based extractive graph-based approach for summarizing Arabic text. The original text was converted into a graph using this suggested strategy. The sentences were represented as vertices, and the linkages between the sentences were determined using the mutual nouns between the connected phrases and the cosine similarity between the sentences based on Term Frequency-Inverse Document Frequency (TF-IDF).

The extractive Arabic text summarizing approach proposed by [24] employed the Firefly algorithm. The proposed approach comprised four basic steps: (1) text preprocessing techniques such as segmentation, tokenization, stop word elimination, and stemming; (2) using a phrase's structural features, such as the title similarity, sentence length, sentence placement, and term TF-IDF weight, to calculate similarity scores; (3) creating a graph of potential answers, where the vertices are the original document's sentences, and the edges are how close they are to one another; (4) choosing which sentences should be in the summary using the Firefly algorithm. The suggested method was assessed using the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metrics on the EASC corpus.

The QUESTS system, which was suggested in [25,26], is an integrated query system for producing extractive summaries from a collection of documents. In order to create many subgraphs from the main graph, this system first creates an integrated graph of the relationships between the sentences of all the input documents. Sentences that are more closely related to the topic at hand and to one another make up these subparagraphs. The highest-scoring subparagraph that is most pertinent to the query is chosen for inclusion in the summary after the algorithm ranks the subparagraphs using a scoring model.

3. The Roots of Arabic Words

The roots of words are one of the Arabic language's strengths. Arabic words typically have a root, which means that the root can serve as the foundation for other words with similar meanings. By adding suffixes to the root, we can create a set of derivations. These derivatives cover the same ground. Finding an Arabic word's root (also known as stemming) facilitates the mapping of grammatical differences to instances of the same term.

Multi-derivations of the wording structures in the Arabic language allow a semantic representation of the text that is closer to the semantic foundations. For instance, the root "درس" is used for many words relating to "reading", including " ' .دارس" and "مدرسة". It is worth noting that it is a difficult matter to determine the root of any Arabic word because of that.

Due to a variety of factors, the Arabic language has been regarded as difficult for automatic text summarization and information retrieval. Because Arabic words can take on a wide variety of forms and is a highly inflectional and derivational language, studying morphology can be exceedingly challenging. Additionally, the way a character is written depends on where the letter falls within a word, which might make it more difficult to analyze Arabic words. Therefore, for the Arabic language, obtaining the lemma, stem, or root is a challenging problem. Based on such Arabic language standards, natural language processing appears more complex and takes more time than what has been achieved in English and other European languages.

The quality and accuracy of the task of artificial text summarization may be positively impacted by a good representation of Arabic text. Additionally, as words with the same root are semantically connected, feature selection methods based on the root can enhance a method for determining how similar two passages of Arabic text are, which can be applied as the foundation for our Arabic text summarizing strategy.

4. Arabic Text Representation for Automatic Summarization Using Graphs

Different Natural Language Processing problems have recently been effective in using graph-based methods. There is a fairly solid mathematical foundation for term significance determination techniques. The approach of determining a textual unit's relevance has become increasingly popular in graph-based ranking algorithms. It is possible to determine the relative relevance of a node within the graph using graph-based ranking algorithms. When determining the significance of a node, these algorithms consider the global information, or the entire graph, rather than just the local, vertex-specific information. Sentences or other text elements are connected by meaningful relations in a text that is represented as a graph. We will be better able to understand the relationship between the various components of the text by using the graph to depict the text's organizational structure. The different sections of a text are ranked using graph-based methods, where each piece is treated as a node. The lexical or semantic relationships between two nodes will be represented by edges. It is possible to connect two graph vertices by drawing an edge between them, representing, for instance, lexical or semantic relationships. No matter the nature or qualities of the text we wish to graph, a graph-based ranking system must first perform the following basic steps:

1. Determine which text units—sentences, words, or other units—best describe the assignment and use them as nodes in a graph.
2. Identify the relationships that link these text units, then utilize those relationships to create edges between the graph's vertices. Edges may be weighted or unweighted, directed or undirected.
3. Until convergence, use the graph ranking algorithm to determine a ranking over the graph's nodes. Then, every node is arranged in order of ultimate score. Use the values associated with each vertex to determine ranking and selection.

As indicated in the third phase, nodes are ranked according to their final scores after specifying the final scores for each node. The best sentences are then chosen to participate in the final summary. Two of the most significant algorithms based on the graph are TexRank [27] and LexRank [28]. We then briefly looked at each of these algorithms.

The candidate sentences that might be included in the summary are all represented in a graph by the multi-document summarizing system called LexRank. If the similarity between two sentences exceeds a certain level, they are connected in this graph representation. A competitive advantage between two sentences is created if they have specific similarities. This similarity is computed using the function COSINUS. The system then conducts a random walk on the graph after constructing the network to identify the most crucial sentences.

All graphs that are derived from natural language texts are ranked using the graph-based model TextRank.

TextRank is a single document summarizing system that derives from the Google page ranking [27,29] paradigm. Keywords and sentences are extracted using TextRank. To extract sentences, a completely connected, undirected graph is used. An individual sentence is regarded as a vertex (or node) in a graph. A similarity connection that is calculated as a function of shared concepts is used to draw a line between two texts. Additionally, each edge has a weight that reflects how significant a relationship is. The best sentences are chosen after ranking each sentence according to its score.

Given a document d, let $G = (V, E)$ be an undirected graph that represents the document d with the set of nodes V and the set of edges E. This is for the text summarizing task. The nodes in this model stand in for the sentences in d. Every edge E_{ij} has a weight W_i that denotes how similar the nodes (sentences) V_i and V_j are to one another. If two sentences satisfy a similarity threshold t and are similar to one another, then they are related. Based on the relationships with other connected nodes, each node in the V graph is also given a salient score. This score, which was determined using a ranking system, shows how much information is contained in a sentence.

5. Proposed Arabic Summarization Method

In order to effectively summarize text, graph-based ranking algorithms have also been demonstrated to be useful. Each sentence in the text is added as a vertex, and the edges between the vertices are made up of connections made by other sentences. These relationships are established by the use of a similarity relation, where similarity is determined by the degree of content overlap. In this study, we demonstrate the outcomes of using graph theory for the summary of Arabic text. Figure 1. bellow shows the overview of the proposed approach for triangle-graph based summarization system.

The five main steps of this approach are as follows:

1. Data Preprocessing;
2. Text Graph-based Representation;
3. Sub-graph construction;
4. Sentence ranking;
5. Summary generation.

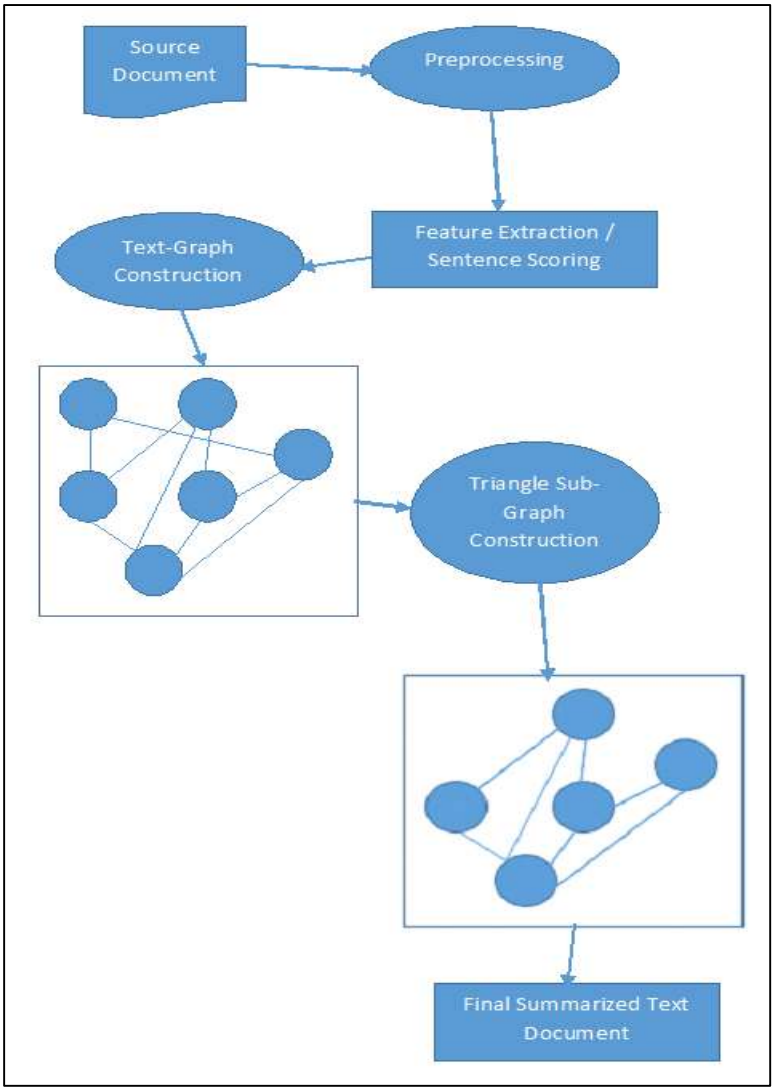

Figure 1. Overview of the proposed approach for triangle-graph-based summarization.

5.1. Data Pre-Processing

It is challenging to test and evaluate an artificial text summarizing system since there is no perfect summary for almost any specific document or set of related texts. Additionally, as researchers typically gather their own information, the lack of Arabic standard datasets made the evaluation process more difficult and possibly subjective in some circumstances [30]. As far as we are aware, there are four Arabic extractive single-document datasets that are available to the public. Summaries are produced automatically by translating an English corpus into Arabic using Google's translation service. When compared to human translation, this method of dataset generation lowers the cost of creating an Arabic dataset. However, doing so could result in a document of poor quality or have an impact on semantics. To automatically produce extractive summaries that might be biased toward certain summarizers, authors in [31] previously built Arabic summarizers. Finally, the dataset in [15] was created using human-generated extractive summaries. In order to test and assess the suggested strategy, the Essex Arabic Summaries Corpus (EASC) [15] has been used. A team of scholars at Essex University created the EASC corpus, an extraction summation that was published. It has 153 articles on various subjects that were compiled from Arabic newspapers and Wikipedia. There are five separate reference summaries produced by five different humans for each article in the EASC corpus. The one thing that sets this dataset apart from others is that it is the only Arabic dataset that

has been created by humans. This makes the evaluation more realistic when compared to methods that rely on translated datasets or the output of summarizers that have already been produced.

The first stage in practically all summary methodologies is this one. Its major objective is to get the input text file ready for processing in subsequent phases. It primarily creates a uniform representation of the input document.

Due to the complexity of the Arabic language, developing the NLP system is not simple. The rich and intricate morphological and syntactic flexibility of Arabic is widely known [32]. The preprocessing stage is essentially the same for all languages and often entails normalization, tokenization, POS tagging, stemming/lemmatization, and stop-word removal [33–35]. Since most texts produced in Arabic and saved in electronic form do not have diacritical marks at first, the system deals with Arabic texts without them.

5.1.1. Tokenization

Tokenization, the first step in text preprocessing, divides input documents into units of varying levels to make it easier to access all of the input document's content. These units can be tokens, sentences, paragraphs, numerals, or any other suitable unit [36]. To give an example, the proposed tokenization is a morphological decomposition based on punctuation that begins by identifying the paragraphs the document is made up of. The newline character *n* serves as the paragraph divider in this scenario. Following that, paragraphs are divided into a collection of phrases using the full stop (.), question mark (?), and exclamation mark (!). Finally, delimiters such as white space, semicolons, commas, and quotations are used to separate these phrases into tokens. To handle the aforementioned series of actions, we used the AraNLP tool with minimal modification [37].

5.1.2. Normalization

Some Arabic letters may take on several forms, while others may be used in place of others because of similarities in their shapes. Writers also employ diacritical marks in their writing. These result in a set of variations for the same term, which has an impact on how some attributes, such as term frequency (*TF*), are computed. To avoid these variations, a normalization technique is needed to harmonize the many spellings of the same letter. The following activities are performed by the suggested normalization step using the AraNLP tool [37]: (i) eliminating non-Arabic letters such as special symbols and punctuation; (ii) removing diacritics; (iii) replacing آ ،أ and إ with ا, ى with ي, and ة with ه [38]; (iv) removing tattoos (stretching characters).

5.1.3. Stop Words Removal

Stop words are unimportant words that regularly appear in texts to build sentences, such as pronouns, prepositions, conjunctions, etc. [39]. These words can be removed from sentences without changing their main ideas since they are not informative (do not add information). In fact, this phase is very important because several computations are based on the frequency of the words in the sentence or document. Therefore, by eliminating stop words, these calculations are made more pertinent and precise. Stop-words are eliminated from the text using a variety of stop-list techniques, including the general stop-list, corpus-based stop-list, and combined stop-list. The suggested strategy, which outperformed the other two ways, relies on a broad stop-list created with the AraNLP tool [37,40].

5.1.4. Stemming

Because Arabic is a highly inflectional and derivational language, words can take on a wide variety of forms while still having the same action-related abstract meaning. Evidently, this has an impact on a number of natural language processing techniques, including text similarity analysis and developing bag-of-word models. Stemming, then, is the process of deleting all or some affixes from a word, such as prefixes, infixes, postfixes, and suffixes. In

other words, stemming reduces a word's various forms and derivatives to a single, unified form (such as a root or stem) from which all other forms can be derived. There are two popular stemming techniques in Arabic: light stemming and morphological root-based stemming [41]. When comparing these methods for text summarization, authors in ref. [42] used three well-known Arabic stemmers, namely the Khoja root stemmer. Their research showed that root stemming outperforms light stemming for summarizing Arabic texts. We modified a Khoja root stemmer to handle the stemming operation as a preprocessing task for the proposed study based on those findings.

5.1.5. Feature Extraction

The set $D = (S_1, S_2, \ldots, S_k)$ represents the textual document, with S_1 being a phrase from document D. The textual contents are then subjected to feature extraction, and helpful primary sentence and word structures are identified. Each document includes a variety of structural elements, including title words, sentence lengths, sentence positions, numerical data, term weights, sentence similarity, thematic-word and proper-noun instances, and sentence lengths and positions.

Title words: Sentences containing title words that accurately reflect the meanings of the arguments are given higher ratings. The following method is used to determine this:

$$TF(S_i) = \frac{CountWord(S_i) \cap CountWord(Title)}{CountLength(Title)} \quad (1)$$

Sentence lengths: Lines including the date or author are eliminated from sentences that are too short. The normalized length of each sentence is calculated as:

$$SL(S_i) = \frac{CountLength\left(S_{(i,w\in\{1\ldots n\})}\right)}{CountLength\left(S_{(j,w\in\{1\ldots m\})}\right)} \quad (2)$$

Sentence positions: Sentences that appear earlier in their paragraphs are given higher grades. Each sentence in a paragraph with n sentences is scored as follows:

$$SP(S_i) = \frac{CountTotal(d) - CurrentPosition(S_i)}{CountTotal(d)} \quad (3)$$

Numerical data: Each sentence containing numerical terms that duplicate significant statistical data points in the text is slated for summarization. The scores for each phrase are calculated as follows:

$$ND(S_i) = \frac{CountND(S_i)}{CountLength\left(S_{(i,w\in\{1\ldots n\})}\right)} \quad (4)$$

Thematic words: The number of thematic words, or domain-specific phrases exhibiting the highest level of relativeness, found in a sentence divided by the number of thematic words found in the sentences, is calculated as follows:

$$TW(S_i) = \frac{CountThematic(S_i)}{max(TW)} \quad (5)$$

Sentences that are identical to one another: To determine commonalities between each sentence S and every other sentence, token-matching algorithms are used. The total number of sentences found is represented by the matrix [N][N], and the diagonal components are set to zero because the sentences are not compared to one another. The evaluation of each sentence's similarity score is as follows:

$$STS(S_i) = \frac{\sum_{k=1}^{n} Sim(S_i, S_j)_k}{max(sim(S_i, S_j)_k)} \quad (6)$$

5.1.6. Similarity Measuring

One of the most widely used similarity measures for text documents is cosine similarity, which is used in many applications for information retrieval and clustering. Based on the TF/IDF feature, the cosine similarity between two sentences, t_1 and t_2, is as follows:

$$SIM(t_1, t_2) = \frac{\sum_{i=1}^{n} t_{1i} t_{2i}}{\sqrt{\sum {t_{1i}}^2} \times \sqrt{\sum {t_{2i}}^2}} \quad (7)$$

5.2. Text Graph Representation

A text is divided into sentences and words before being summarized. This stage involves formatting an Arabic text document as a graph. The collection of vertices V and the set of edges E that represent the document are created to form the undirected weighted graph $G = (V, E)$. The sentences act as the graph's nodes. When two sentences are similar to one another, they have an edge between them. The edges of the graph show this similarity, and the edge weight indicates how similar the phrases are. Many other approaches can be used to determine how similar two sentences are in Arabic text, including Cosine similarity, Jaccard, Word-Overlap, and dice. We employ the cosine similarity measure in this study. If the similarity between two sentences exceeds a predetermined threshold ($t = 0.5$ in the trials), the sentences are considered connected. This process produces a graph that is extremely linked. The link between the two sentences that each edge connects is represented by its edge. The edge weight represents how well the sentences in the paper are connected to one another. This undirected weighted graph serves as the input for the procedure used to determine each sentence's salient points in the following section.

The sentences in a text will be ranked using random walk on G once the document graph has been constructed. Using the PageRank technique, we get the salience score for each node [43]. PageRank was created as a mechanism for Web link analysis and is one of the most well-known link analysis algorithms. Using data from the graph's structure, it assesses a node's significance within the network. Although PageRank was designed to be used with directed graphs, it can also be effective with undirected graphs. By doing this, a vertex's output-degree and input-degree are equal. In our case, $In(V_i)$ equals Out because the graph is undirected V_i. The score of a vertex V_j is given by Equation (8), where $In(V_i)$ is the set of nodes that point to VI'. $Out(j)$ is the set of nodes that node j points to, W_{ij} is the weight of the edge leading from node V_j to node j, and d is a damping factor that can be set between 0 and 1. This damping factor serves to incorporate into the model the probability of jumping from a given vertex to another random vertex. Typically, the value of d is 0.85 [27].

$$PR(V_i) = (1 - d) + d * \sum V_{j \in \ln(V_i)} w_{ij} \frac{PR(V_j)}{\sum V_{k \in Out(V_j) W_{jk}}} \quad (8)$$

To determine PR, a starting score of 1 is given to each node, and Equation (8) is applied iteratively on the weighted graph G until the difference in scores between iterations for all nodes is less than a threshold of 0.001. The salient scores of the sentences determine the nodes' weights. Nodes with higher scores correspond to sentences that are significant, relevant to the document, and have strong relationships with other sentences. Each vertex is given a score following the algorithm's execution, and this score reflects the vertex's "importance" or "power" inside the graph. After that, the sentences are arranged in order of their scores. Note that only the number of iterations necessary to reach convergence may be impacted by the initial value choice; the final values are unaffected. Figure 2. shows a graph representation built for a text sample, the blue lines are the edges between sentences of the text.

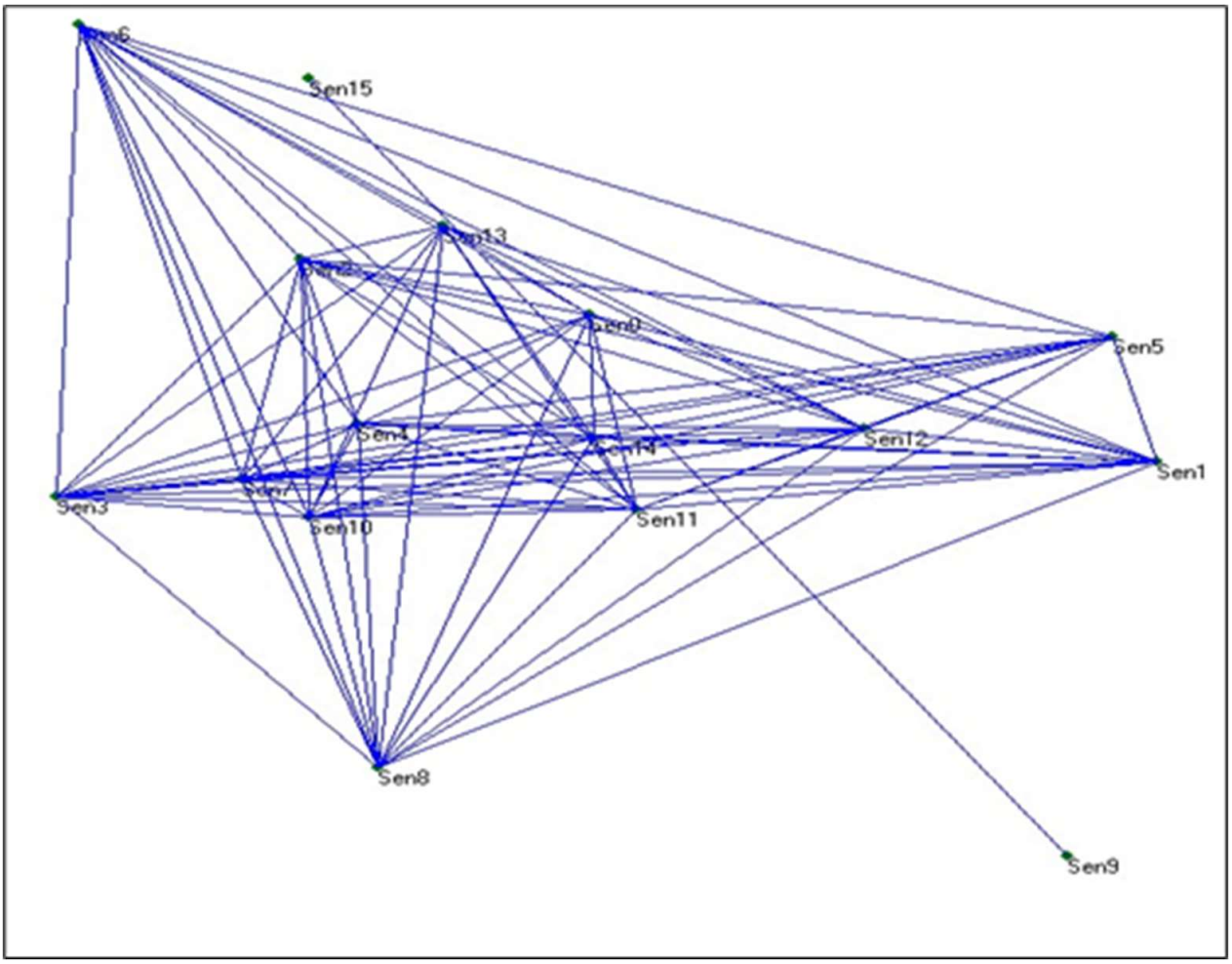

Figure 2. Sample graph built for text representation.

5.3. Sub-Graph Construction

The triangular sub-graph construction process comes next. Triangles use the axiom that people who know people who know people tend to be friends. We start by making an adjacency matrix. Algorithm 1. shows how the adjacency matrix works.

Algorithm 1: Adjacency Matrix
Input: Graph Data Set with N nodes or no of sentences and E edges or relationships between sentences
Output: N*N Adjacency Matrix, showing the connections between nodes
Start
Determine the size of the matrix, which is N*N (N is the number of nodes in the graph)
Create the matrix A
For each node v ∈ N {
If there is any edge from v_i to v_j {
A (i,j) = 1 —if sim (Si, Sj) > 0
Else
A (i,j) = 0 —if there is there is no words that are similar between sentences
}
Stop

The next step is to create a list of triangles to represent the text. The procedure based on De- Morgan lows is used to locate the triangles in the graph. Algorithm 2. shows how this step was done.

Algorithm 2: De-Morgan lows
Input: N*N adjacency matrix, (A(I,J))
Output: Array of triangles
Start
Triangles_Array = [],
For each edge in the matrix A(I,J), namely XY, find all edges start with Y {
$XY \wedge YZ \longrightarrow XZ$
If $XZ \in A(I,J)$ {
Add the triangle of edges (X,Y,Z) to Triangles_Array[]}
}
Stop

After finding the nodes and edges representing the triangles in the main graph, we can construct the reduced graph. Figure 3 shows the Triangle graph for the sample text represented in Figure 2. The blue lines are the edges between sentences of the text, while the red lines are the edges represent the reduced graph. That means only the red edges will be used to create the summary.

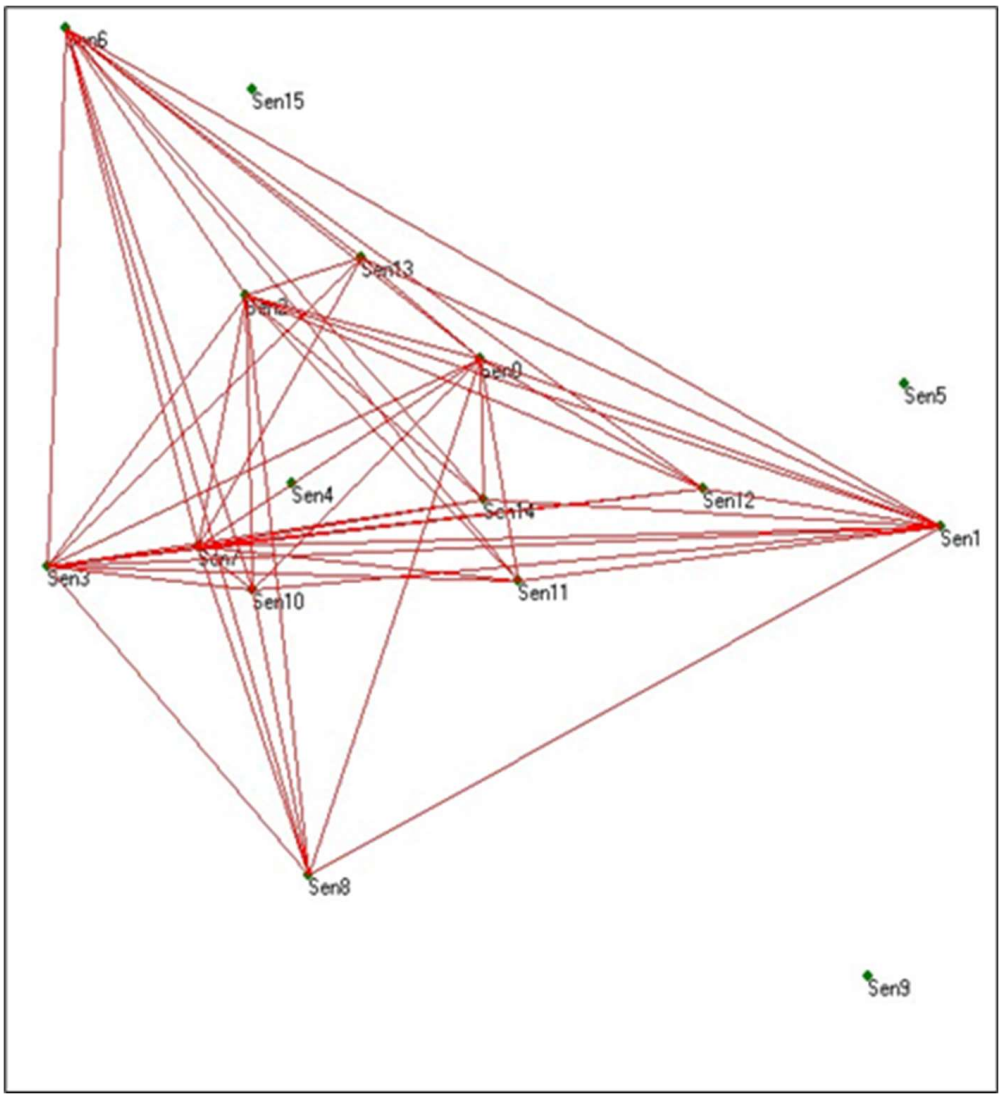

Figure 3. Triangle graph-based text representation.

In Figure 3, the red lines show the edges of the triangles from the main graph, while the blue ones were ignored.

5.4. Sentence Scoring

To find the most vital sentences, the Bit–vector exemplification was adopted in this work to symbolize the pruned graph from the preceding section. Each sentence has either one feature or multi-features. In our work, we used a combination of the six features discussed above. The combinations could be two, three, four, five, or six features. We had 63 probabilities for these combinations.

After scoring all six features explained above, a principle statistics method was used to construct a document summary. Text summarization based on general statistics methods was exploited to integrate the six feature scores combined with bit-vector values as the sentence weight.

After features were extracted by the system, the sentence scores were obtained. First, a weighted score function for a sentence S is exploited to integrate all six features, as calculated using Equation (9).

$$Score\ (S_i) = Bit_{Vectore}(S_i) * \sum_{k=1}^{m} Score(F_k(S_i)) \quad (9)$$

where $Score(S_i)$ is the score of sentence S, $Score(F_k(S_i))$ is the score of feature K, and m is the number of features used to score the sentences.

5.5. Summary Generation

Each sentence in the manuscript was given a value based on the sentence scores acquired. Only sentences with sub-graph structures were chosen for analysis since they are connected to at least two additional sentences. According to its grade, each sentence was ranked in decreasing order. High-scoring candidates were removed for document summarizing, in accordance with the compression ratio. It has been shown that an extraction or compression rate of close to 20% of the core textual material is just as informative of the contents as the full text of the document [44]. The summary sentences in the last step are arranged according to the order of the sentences in the original text.

6. Experimental Results

The proposed experiment aims to produce the following outcomes: (i) assess the proposed design of the chosen statistical and semantic features; (ii) assess the use of a statistical summarization method on the Arabic texts; and (iii) assess the comparison of our proposed method to other related works. As was already noted, the EASC dataset was used for testing and assessing the suggested method. In order to calculate the precision, recall, and F-score for each of the generated summaries for both summary methods, ROUGE-N (i.e., ROUGE-1, ROUGE-2) was employed.

In order to produce the output summaries in score-based summarization, an input threshold (summary ratio) needs to be modified. Finding the ideal ratio is challenging because the corpus includes 153 documents, each of which has five human reference summaries with a different ratio. To prevent this issue, the generated summaries are modified using an adaptive ratio dependent on the length of the reference summary we are comparing it to.

A majority summary, or so-called gold-standard summary, was created by a voting process among the five references to improve outcomes and avoid the problem of subjectivity. As a result, the statement was included in the gold-standard reference summary if it appears in three or more of the five references [45].

The results of the suggested approach are contrasted with those of other systems and methods for relevant Arabic summarization in this section. With a brief description of the summary type, summarization method, and features employed, Table 1 presents ten similar summarization methods/systems. These systems were assessed using the Essex Arabic corpus and the "gold-standard" summary, which stipulates that no more than 50% of the original document's words should be used in the summary.

Table 1. Performance evaluation compared with other research.

System	Recall	Precision	F-Measure
Al-Radaideh and Afif (2014) [46]	0.161	0.191	0.175
Haboush et al. (2012) [47]	0.18	0.22	0.198
LCEAS (AL-Khawaldeh and Samawi, 2015) [48]	0.271	0.293	0.282
mRMR (Oufaida et al., 2014) [49]	0.282	0.327	0.303
AQBTSS (El-Haj et al., 2009) [14]	0.445	0.493	0.468
LSA-Summ (El-Haj et al., 2009) [14]	0.605	0.417	0.494
Gen-Summ (El-Haj et al., 2009) [14]	0.599	0.488	0.518
ESMAT (Binwahlan, 2015) [50]	0.589	0.488	0.518
Al-Radaideh and Bataineh (2018) [51]	0.465	0.376	0.422
Al-Abdallah (2017) [52]	0.449	0.482	0.524
Proposed graph based	0.633	0.601	0.617

Since ROUGE-N (N = 2) performs better for the evaluation of single document summarization, it was employed in the evaluation process as an automatic evaluation metric for recall, precision, and F-score. Based on their published results in terms of recall, precision, and F-Score, Figure 4 compares the performance results of the proposed summarizing method to the performance results of the related summarization methods/systems. The suggested score-based strategy exceeds the competition in terms of recall, precision, and F-Score, with average improvements of 23%, 23%, and 24%, respectively (Figure 4). This is due to the potency/strength of the chosen feature and the originality of their composition, in addition to the use of appropriate and modern Arabic NLP techniques.

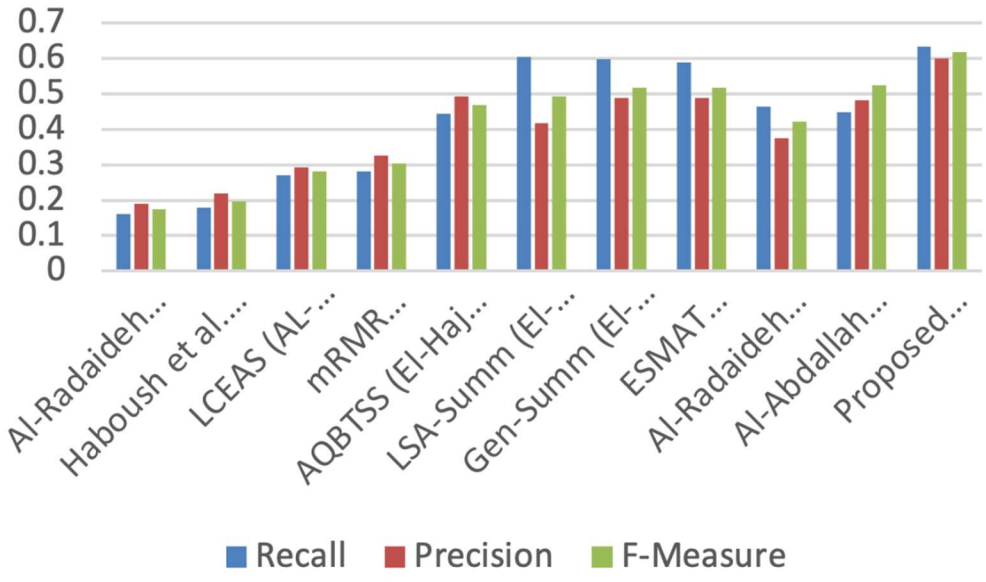

Figure 4. Performance evaluation compared with other research.

7. Discussion

The experiment results of the proposed method based on the triangle sub-graph using cosine similarity measurement and specific selected features show that the resulting summaries could be better than other summaries. DUC 2002 was used as a data warehouse for news article collection as input in our experiment. Three pyramid evaluation metrics (mean coverage score (recall), average precision, and average F-measure) are employed for the comparative evaluation of the proposed approach and other summarization systems. In

this approach, we used six different features for each sentence, and we used cosine similarity measurement to find the relations between the sentences (graph nodes) to represent the graph; then we pruned the graph by finding the triangles sub-graph and by using the sentences, formed this sub-graph to find the summary. The scoring process of the sentences was completed based on the values of the selected features. Based on the experimental results of the proposed method, we can say that if we can identify significant similarity measurements for representing relations between sentences and identify significant features for text summarization, it can produce a good summary.

8. Conclusions

Because of the Internet's incredible rise in data, it is more important than ever to have an automated summarizing system that can reduce user time consumption and information overload. Key sentences from the document's major concepts should be retained in a decent summary, and repetition should be minimized to create a summary that is information-rich. Despite current efforts to develop text summarization techniques and formulate representative characteristics, these formulations are still unable to adequately capture the relevance, coverage, and diversity of a phrase. The method for extracting single document summarization presented in this paper is general.

The score-based method makes use of a set of attributes that were selected and developed after a thorough examination of summarization techniques, Arabic text characteristics, and writing styles. These characteristics range from statistics to semantically-based ones. While keeping in mind that these sentences are varied and cover the entire notions of the document, the adopted formulations aid in determining the value of sentences, which is vital to the process of deciding whether to include them in the summary. We test the suggested strategy using the EASC dataset. The system achieved an F-score of 0.617 for the score-based method using ROUGE-2 as a performance metric.

The findings obtained demonstrate that our method outperforms the most cutting-edge score-based algorithms, particularly in terms of precision. This is a result of the proposed characteristics' informative formulation, which aids in highlighting the significance of the statement.

Author Contributions: Conceptualization, Y.A.A.-K. and E.S.H.; Methodology, Y.A.A.-K. and E.S.H.; Software, Y.A.A.-K. and E.S.H.; Validation, Y.A.A.-K. and E.S.H.; Formal analysis, Y.A.A.-K. and E.S.H.; Investigation, Y.A.A.-K. and E.S.H.; Resources, Y.A.A.-K. and E.S.H.; Data curation, Y.A.A.-K. and E.S.H.; Writing—original draft, Y.A.A.-K. and E.S.H.; Writing—review & editing, Y.A.A.-K. and E.S.H.; Visualization, Y.A.A.-K. and E.S.H.; Supervision, Y.A.A.-K. and E.S.H.; Project administration, Y.A.A.-K. and E.S.H.; Funding acquisition, Y.A.A.-K. and E.S.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Deanship of Research in Zarqa University/Jordan. Grant Number: 7252.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Slamet, C.; Atmadja, A.R.; Maylawati, D.S.; Lestari, R.S.; Darmalaksana, W.; Ramdhani, M.A. Automated text summarization for indonesian article using veSctor space model. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *288*, 012037. [CrossRef]
2. Hosseinikhah, T.; Ahmadi, A.; Mohebi, A. A new Persian text summarization approach based on natural language processing and graph similarity. *Iran. J. Inf. Process. Manag.* **2018**, *33*, 885–914.
3. Ozsoy, M.G.; Alpaslan, F.N.; Cicekli, L. Text summarization using latent semantic. *J. Inf. Sci.* **2011**, *37*, 405–417. [CrossRef]
4. El-Kassas, W.S.; Salama, C.R.; Rafea, A.A.; Mohamed, H.K. Automatic text summarization: A comprehensive survey. *Expert Syst. Appl.* **2021**, *165*, 113679. [CrossRef]
5. Talibali, L.; Riahi, N. An overview of automatic text summarization techniques. *Int. J. Eng. Res. Technol.* **2015**, *28*, 75–84.
6. Thakkar, K.; Dharaskar, R.; Chandak, M. Graph-Based Algorithms for Text Summarization. In Proceedings of the 2010 3rd International Conference on Emerging Trends in Engineering and Technology (ICETET), Goa, India, 19–21 November 2010; pp. 516–519.
7. Luhn, H.P. The automatic creation of literature abstracts. *IBM J. Res. Dev.* **1958**, *2*, 159–165. [CrossRef]

8. AL-Khassawneh, Y.A. The use of Semantic Role Labelling with Triangle-Graph Based Text Summarization. *Int. J. Emerg. Trends Eng. Res.* **2020**, *8*, 1162–1169. [CrossRef]
9. Belwal, R.C.; Rai, S.; Gupta, A. A new graph-based extractive text summarization using keywords or topic modeling. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 8975–8990. [CrossRef]
10. Li, Y.; Cheng, K. Single document Summarization based on Clustering Coefficient and Transitivity Analysis. In Proceedings of the 10th International Conference on Accomplishments in Electrical and Mechanical Engineering and Information Technology, Banjaluka, Srpska, 26–28 May 2011.
11. Mihalcea, R. Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, 21–26 July 2004.
12. AL-Khassawneh, Y.A.; Salim, N.; Jarrah, M. Improving triangle-graph based text summarization using hybrid similarity function. *Indian J. Sci. Technol.* **2017**, *10*, 1–15. [CrossRef]
13. AL-Khassawneh, Y.A.; Salim, N.; Isiaka, O.A. Extractive text summarisation using graph triangle counting approach: Proposed method. In Proceedings of the 1st International Conference of Recent Trends in Information and Communication Technologies in Universiti Teknologi Malaysia, Johor, Malaysia, 12–14 September 2014; pp. 300–311.
14. El-Haj, M.; Kruschwitz, U.; Fox, C. Experimenting with Automatic Text Summarization for Arabic. In Proceedings of the 4th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics, LTC'09, Poznan, Poland, 6–8 November 2009; pp. 365–369.
15. El-Haj, M.; Kruschwitz, U.; Fox, C. Using Mechanical Turk to Create a Corpus of Arabic Summaries in the Language Resources (LRs) and Human Language Technologies (HLT) for Semitic Language. In Proceedings of the Workshop Held in Conjunction with the 7th International Language Resources and Evaluation Conference (LREC), Valletta, Malta, 17–23 May 2010; pp. 36–39.
16. Ben Abdallah, M.; Aloulou, C.; Belguith, L. Toward a Platform for Arabic Automatic Summarization. In Proceedings of the International Arab Conference on Information Technology (ACIT'08), Hammamet, Tunisia, 16–18 December 2008.
17. Sakhr Company. Available online: http//:www.sakhr.com (accessed on 1 October 2022).
18. Boudabous, M.; Maaloul, M.; Belguith, L. Digital learning for summarizing Arabic documents. In Proceedings of the 7th International Conference on NLP (IceTAL 2010), Reykjavik, Iceland, 16–18 August 2010.
19. Sabuna, P.M.; Setyohadi, D.B. Summarizing Indonesian text automatically by using sentence scoring and decision tree. In Proceedings of the 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, Indonesia, 1–2 November 2017; Volume 9, pp. 1–6.
20. Abuobieda, A.; Salim, N.; Albaham, A.T.; Osman, A.H.; Kumar, Y.J. Text summarization features selection method using pseudo genetic-based model. In Proceedings of the 2012 International Conference on Information Retrieval & Knowledge Management, Kuala Lumpur, Malaysia, 13–15 March 2012; Volume 8, pp. 193–197.
21. Chowdary, C.R.; Sravanthi, M.; Kumar, P.S. A system for query specific coherent text multi-document summarization. *Int. J. Artif. Intell. Tools* **2010**, *19*, 597–626. [CrossRef]
22. Thaokar, C.; Malik, L. Test model for summarization Hindi text using extraction method. In Proceedings of the 2013 IEEE Conference on Information & Communication Technologies, Thuckalay, India, 11–12 April 2013; Volume 7, pp. 1138–1143.
23. Elbarougy, R.; Behery, G.; Khatib, A.E. Graph-Based Extractive Arabic Text Summarization Using Multiple Morphological Analyzers. *J. Inf. Sci. Eng.* **2020**, *36*, 347–363.
24. Al-Abdallah, R.Z.; Al-Taani, A.T. Arabic Text Summarization using Firefly Algorithm. In Proceedings of the 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates, 4–6 February 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 61–65.
25. Riahi, N.; Ghazali, F.; Ghazali, M. Improving the efficiency of the Persian abstract synthesis system using pruning algorithms in neural networks. In *Proceedings of the First International Conference on Line and Language Processing Persian*; Semnan University: Semnan, Iran, 2012.
26. Shafiei, F.; Shamsifard, M. The automatic dictionary of Persian texts. In Proceedings of the 20th National Computer Society Conference, Mashhad, Iran, 3 March 2015; Volume 1, pp. 931–936.
27. Mihalcea, R.; Tarau, P. Textrank: Bringing order into text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain, 25–26 July 2004.
28. Erkan, G.; Radev, D.R. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.* **2004**, *22*, 457–479. [CrossRef]
29. Brin, S.; Page, L. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **1998**, *30*, 107–117. [CrossRef]
30. Al-Saleh, A.; Menail, M. Automatic Arabic text summarization: A survey. *Artif. Intell. Rev. Arch.* **2016**, *45*, 203–234. [CrossRef]
31. El-Haj, M.; Koulali, R. Kalimat a multipurpose Arabic corpus. In Proceedings of the 2nd Workshop on Arabic Corpus Linguistics (WACL-2), Lancaster, UK, 22 July 2013.
32. Attia, M. Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation. Ph.D. Thesis, School of Languages, Linguistics and Cultures, Faculty of Humanities, University of Manchester, Manchester, UK, 2008.

33. Abdelkrime, A.; Djamel Eddine, Z.; Khaled Walid, H. Allsummarizer system at multiling 2015: Multilingual single and multi-document summarization. In Proceedings of the SIGDIAL 2015 Conference, Prague, Czech Republic, 2–4 September 2015; pp. 237–244.
34. Litvak, M.; Vanetik, N.; Last, M.; Churkin, E. Museec: A multilingual text summarization tool. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics—System Demonstrations, Berlin, Germany, 7–12 August 2016; pp. 73–78.
35. Thomas, S.; Beutenmüller, C.; de la Puente, X.; Remus, R.; Bordag, S. Exb text summarizer. In Proceedings of the SIGDIAL 2015 Conference, Prague, Czech Republic, 2–4 September 2015; pp. 260–269.
36. Attia, M. Arabic tokenization system. In Proceedings of the 5th Workshop on Important Unresolved Matters, Prague, Czech Republic, 28 June 2007; pp. 65–72.
37. Althobaiti, M.; Kruschwitz, U.; Poesio, M. Aranlp: A java-based library for the processing of Arabic text. In Proceedings of the Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, 26–31 May 2014.
38. Ayedh, A.; Tan, G.; Alwesabi, K.; Rajeh, H. The effect of preprocessing on Arabic document categorization. *Algorithms* **2016**, *9*, 27. [CrossRef]
39. Al-Shalabi, R.; Kanaan, G.; Jaam, J.M.; Hasnah, A.; Hilat, E. Stop-word removal algorithm for Arabic language. In Proceedings of the 2004 International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, 23 April 2004.
40. El-Khair, I. Effects of stop words elimination for Arabic information retrieval: A comparative study. *Int. J. Comput. Inform. Sci.* **2006**, *4*, 119–133.
41. Mustafa, M.; Salah-Eldeen, A.; Bani-Ahmad, S.; Elfaki, A. A comparative survey on Arabic stemming: Approaches and challenges. *Intell. Inf. Manag.* **2017**, *09*, 39–67. [CrossRef]
42. Alami, N.; Meknassi, M.; Ouatik, S.A.; Ennahnahi, N. Impact of stemming on Arabic text summarization. In Proceedings of the 2016 4th IEEE International Colloquium on Information Science and Technology (CiSt), Tangier, Morocco, 24–26 October 2016.
43. Brin, S.; Page, L. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw.* **2012**, *56*, 3825–3833. [CrossRef]
44. Morris, A.H.; Kasper, G.M.; Adams, D.A. The effects and limitations of automated text condensing on reading comprehension performance. *Inf. Syst. Res.* **1992**, *3*, 17–35. [CrossRef]
45. El-Haj, M. Multi-Document Arabic Text Summarisation. Ph.D. Thesis, University of Essex, Colchester, UK, 2012.
46. Al-Radaideh, Q.; Afif, M. Arabic text summarization using aggregate similarity. In Proceedings of the 2009 International Arab Conference on Information Technology (ACIT'2009), Sana'a, Yamen, 15–18 December 2009.
47. Haboush, A.; Al-Zoubi, M.; Momani, A.; Tarazi, M. Arabic text summerization model using clustering techniques. *World Comput. Sci. Inf. Technol. J.* **2012**, *2*, 62–67.
48. AL-Khawaldeh, F.; Samawi, V. Lexical cohesion and entailment based segmentation for Arabic text summarization (lceas). *World Comput. Sci. Inf. Technol. J.* **2015**, *5*, 51–60.
49. Oufaida, H.; Nouali, O.; Blache, P. Minimum redundancy and maximum relevance for single and multi-document Arabic text summarization. *J. King Saud Univ.-Comput. Inf. Sci.* **2014**, *26*, 450–461. [CrossRef]
50. Binwahlan, M.S. Extractive Summarization Method for Arabic Text-ESMAT. *Int. J. Comput. Trends Technol. IJCTT* **2015**, *21*, 103–107. [CrossRef]
51. Al-Radaideh, Q.A.; Bataineh, D.Q. A hybrid approach for Arabic text summarization using domain knowledge and genetic algorithms. *Cogn. Comput.* **2018**, *10*, 651–669. [CrossRef]
52. Al-Abdallah, R.Z.; Al-Taani, A.T. Arabic single-document text summarization using particle swarm optimization algorithm. *Procedia Comput. Sci.* **2017**, *117*, 30–37. [CrossRef]