*J*ournal of ***AI*** and ***D***ata ***M***ining
Vol 7, No 1, 2019, 121-135

DOI: 10.22044/JADM.2018.6139.1726

A survey on Automatic Text Summarization

N. Nazari and M. A. Mahdavi*

Department of computer engineering, Imam Khomeini International University, Qazvin, Iran.

Received 20 August 2017; Revised 14 November 2017; Accepted 05 February 2018

**Corresponding author: mahdavi@eng.ikiu.ac.ir(M. Mahdavi).*

Abstract

Text summarization endeavors to produce a summary version of a text, while maintaining the original ideas. The textual content on the web, in particular, is growing at an exponential rate. The ability to decipher through such a massive amount of data to extract useful information is a significant undertaking, and requires an automatic mechanism to aid with the extant repository of information. The text summarization systems intent to assist with content reduction keeping the relevant information and filtering the non-relevant parts of the text. In terms of the input, there are two fundamental approaches among the text summarization systems. The first approach summarizes a single document. In other words, the system takes one document as an input and produces a summary version as its output. An alternative approach is to take several documents as its input and produce a single summary document as its output. In terms of output, the summarization systems are also categorized into two major types. One approach would be to extract exact sentences from the original document to build the summary output. An alternative would be a more complex approach, in which the rendered text is a rephrased version of the original document. This paper will offer an in-depth introduction to automatic text summarization. We also mention some evaluation techniques to evaluate the quality of automatic text summarization.

Keywords: *Automatic Text Summarization, Multiple Document Summarization, Single Document Summarization, Summarization Evaluation Technique.*

1. Introduction

With the enormous amount of information generated every day, it is difficult to find the desired information through the manual means. The World Wide Web provides a vast amount of content in forms of web pages, news articles, email, and access to the databases around the world. However, much of this content may not be of use. Automatic text summarization has, therefore, become a research field within the greater field of natural language processing to assist in finding the relevant documents [1]. The goal of automatic text summarization is to extract the main points of the original text without needing to read the entire document. The following definitions form the underlying assumptions regarding text summarization.

"A summary is a text that is produced from one or more texts, and contains a significant portion of the information in the original text(s), and is no longer than half of the original text" [2].

According to Mani [3], "text summarization is the process of extracting the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)".

In terms of the building blocks and the structural components, an automatic text summarization process may be broken into three steps [4, 5]:

- **Identification**: in this step, the main points and central topics of the text are identified. In its simple form, a summary is generated by gathering the important bits of the text.
- **Interpretation**: the important topics that are identified in the first step are combined more cohesively. During this step, some

modifications to the original sentences may prove necessary. Additionally, a successful rendition of the text, in this step, may require domain knowledge.

- **Generation**: the result of the second step is a summary that may not be coherent to the reader. Therefore, the goal of this step is to reformulate the extracted summary into a coherent new text. The generation step is where the final touches of editing are performed to produce an understandable summary for the reader.

2. Various types of summaries

There are different classifications for a summary based on their input, output, purpose, and language [6], which we will elaborate in the following sub-sections.

2.1. Summary based on input

In terms of input, a summary may be based on a single document or multiple documents [7]. Early attempts in summarization were primarily based upon single-document summarization, in which systems produced a summary from a single source document. However, later developments have ushered in text summarizations that are based on multiple source documents. In a multiple-document summarization, several documents that share a similar topic are taken as the input. Given the additional complexity to adjudicate among several documents, the task of multiple-document summarization is proportionally more difficult than the single-document one. This is because the system would have to remove any redundancies across documents and also reconcile the contents into a coherent summary [8].

2.2. Summary based on details

On the basis of its detail, a summary can be either indicative or informative. An indicative summarization system only presents the most important idea of the text. An indicative summary gives an overall perspective of the topics covered by the text. This type of summary helps the user to decide on whether to read the text any further. The typical length of this kind of summary is around 5 to 10 percent of the original text [1, 9]. On the other hand, the informative summarization system covers every aspect of the main text. The length of the informative summaries is around 20 to 30% of the original text [10].

2.3. Summary based on output

Based on the generated text, a summary is either extractive or abstractive [11]. An extractive summary is generated by concatenating the important parts of the text without modifying the original words and sentences. This is a simple and robust way of producing a summary. However, it comes with the risk of producing an inconsistent text since the selected sentences may not share a semantic relation with one another. In other words, an extractive method may generate an incoherent summary [1].
In an abstractive summarization, the natural language generation techniques are used to perform the summarization task [12]. In this method, one tries to understand the original document by identifying the key concepts and then convert it into another semantic form, which amounts to a shorter rephrased version of the original text [11, 13].

2.4. Summary based on content

Based on the content, a summary may be customized according to the needs of the user. In this respect, a summary may be categorized into generic, query-based or domain specific. In a query-based summarization, the summary is generated by selecting a sentence that corresponds to the user's query [14]. The sentences that are relevant to the query receive a higher chance to be extracted for the final summary. The query-based summarization systems, however, do not provide an overall view of the document's concepts since they focus on the user's query. The goal of the generic summarization, on the other hand, is to summarize the whole text without noticing the domain and subject [1]. Generic summaries do not have any view of the topic and consider the document as a unique text, so all the information have the same level of importance [6]. Domain-specific summarizers provide summary according to the specific field [15]. In order to give a few examples, one may refer to summarizing business news articles [16], web pages [17], and biomedical documents [18], amongst many. This kind of summarization requires a domain-specific knowledge to select sentences for summaries.

2.5. Summary based on language

On the basis of the language, there are three types of summaries: monolingual, multi-lingual, and cross-lingual [1, 3]. In a mono-lingual summarization system, the language of source and target documents is the same. FarsiSum is a monolingual text summarization system that produces summary only for Persian text [19]. In a multi-lingual summarization system, the source document and the generated summary could be in some languages. SUMMARIST is a multi-lingual text summarization system that is based upon an extraction method to produce summary from sources in different languages such as English, Indonesian, Spanish, German, Japanese, Korean, and French [20]. The cross-lingual summary is the same as a multi-lingual one but the language of source and target documents must be different.

3. Text summarization method

Much of the efforts made in the field of summarization has been focused on improving the quality of the produced summary. So far, a catalog of different methods has been applied to perform the summarization tasks mentioned in Section 1. In this section, we would like to shed light on various computational methods employed in summarization systems. While the approach to solve the summarization problem may vary from one study to another, the general methods may be categorized into four categories of statistical, machine learning, semantic-based, and artificial intelligent-based methods. Figure 1 gives a hierarchical view of the text summarization methods.

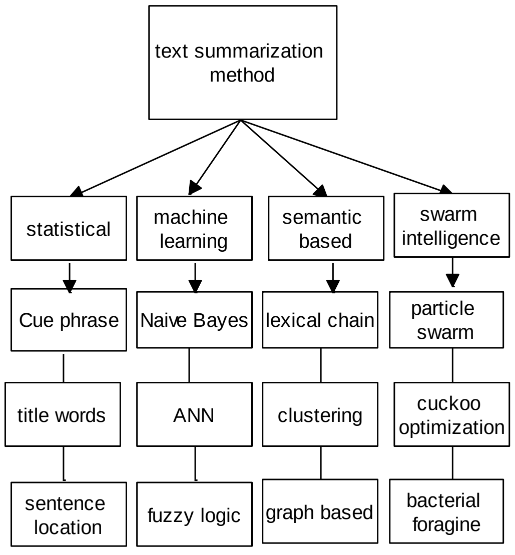

Figure 1: Diagram of text summarization methods

3.1. Statistical method

This method deals with some statistical features of the text to identify the prominent parts of a document. The goal of the statistical method is to select sentences according to the physical features, not the meaning or the relation of words or sentences. Some of the statistical features are word frequency, position of the sentence, and keywords. This section will cover some of these statistical features.

Word frequency is the number of times a word occurs in a text. Luhn [21] has used word frequency for summarizing scientific articles. In this method, Luhn emphasizes that the most frequent words in the text represent the most salient concepts within it. Thus word frequency is used to score the sentences. By comparing scores of each sentence and extracting the sentence with the highest score, the summary is generated.

Baxendale [22] has investigated a machine technique for summarization. He focused on the position of the sentences, and explored that the best location for the important content was the first and last parts of a paragraph. After examining 200 paragraphs, he came to this conclusion that in 85% of them, topic sentences were in the first part of the paragraph, and only in 7% of them they appeared in the last part.

Edmunson [23] has introduced a method for extracting sentences, which not only uses word frequency but also considers the following three features:

1. **Cue phrases**: this feature mentions that the presence or absence of some cue words or phrases such as "as a result," "for example," and "as a matter of fact" indicates the importance of the sentences that include these phrases.
2. **Title words**: this feature predicts that words appearing in the title of a document are immediately relevant to the concepts outlined in the text. Therefore, these words are viewed as a factor to identify the important sentence.
3. **Sentence location**: this feature indicates that the location of a sentence in the paragraph reveals its importance to the topic of the paragraph. For instance, sentences that occur in the initial part of a paragraph carry more information than the rest of the paragraph. Therefore, initial sentences of each paragraph are potentially relevant candidates for sentence selection in a document summary.

The final score is computed by the linear combination of four features. Edmundson examined this method on 400 documents. The results obtained indicated that considering the cue words, title words, and sentence location would produce a higher qualified summary than when only the word frequency feature was applied. In fact, he demonstrated that using word frequency by itself yielded the worst result.

Statistical methods are simple to carry out since they only consider the physical feature of the text. However, these methods do not engage in the meaning of sentences and words, which may produce a low-quality summary.

3.2. Machine learning method

The idea behind machine learning is to use a training set of data to train the summarization system, which is modeled as a classification problem. Sentences are classified into two groups: summary sentences and non-summary sentences [24]. The probability of choosing a sentence for a summary is estimated according to the training document and extractive summaries [25]. Some of the common machine learning methods used for text summarization are naïve Bayes, artificial neural network, and fuzzy logic [26, 27].

3.2.1. Naïve Bayes method

Naïve Bayes is a supervised learning method. In text summarization, the naïve Bayes classification, introduced by Kupiec et al. [28], considers the selection of a sentence as a classification problem. By this classification, each sentence is put in a binary class to determine whether it will be included in the summary or not. The features that are used in this method are word frequency, uppercase words, length of sentence, position in paragraph, and structure of phrase. By considering *k* features and using the Bayes rule, the probability that sentence *s* is included in summary *S* is defined as follows:

$$P(s \in S \mid F_1, F_2, \ldots, F_k) = \frac{\prod_{j=1}^{k} P(F_j \mid s \in S) P(s \in S)}{\prod_{j=1}^{k} P(F_j)} \qquad (1)$$

where, $P(s \in S)$ is a constant, $P(F_j \mid s \in S)$ and $P(F_j)$, are estimated from the training data. Here, a score is assigned to each one of the sentences. Then this score is used towards selecting sentences that would form the summary. Based on this score, the top *n* sentences are extracted, and a summary is produced.

3.2.2. Artificial neural network method

The artificial neural network is a computational model used in computer science and other research areas for solving problems based on machine learning approaches. Kaikhah et al. [29] used artificial neural networks for summarizing news articles as a way to select sentences in an extractive summarization. There are three phases of the proposed approach: neural network training, feature fusion, and sentence selection. The training phase identifies the types of sentences that should be presented in the document summary. A human reader does this, and the system learns the pattern of summary sentences. After training the artificial neural network, the relation among features should be determined. In training the machine, the following seven features are considered:

1. Paragraph follows title
2. Paragraph location in document
3. Sentence location in paragraph
4. First sentence of paragraph
5. Sentence length
6. Number of thematic words in the sentence
7. Number of title words in the sentence

This step consists of two phases: 1) removing uncommon features, and 2) removing the effects of common features. Therefore, this step generalizes the important features that must exist in the summary sentences. After the training and generalizing the network, this system can be used to select important sentences for the summary.

3.2.3. Fuzzy logic method

Fuzzy logic is a multiple-valued logic and an extension of Boolean logic, which was introduced by Lotfi Zadeh [30] for describing the intermediate values between two discrete values like "one" and "zero," and "high" and "low". The advantage of fuzzy logic is the compatibility with the real world, which is not a two-value world. For example, when describing the weather condition, different adjectives such as cold, very cold, warm, hot, and very hot are used in fuzzy logic rather than just using the two value of the Boolean logic. He used the fuzzy logic for the natural language processing, which is called computing with the word. The focus of this research work was to enable the computer to understand the human language that was beyond the concept of one and zero, and could not be implemented by the Boolean logic. In the computer, to process the human language, "the objects of computation are words and propositions drawn from a natural language" [31].

The concept of fuzzy logic can be used in text summarization, which is a branch of natural language processing to help to extract the sentences [32]. Before using fuzzy logic, a pre-processing step was applied to the input text to make the text suitable for the fuzzy logic system that included stopping word removal, stemming, POS tagging, and so on. Then several features of each sentence like title feature, sentence length, and term weight, among others, were considered. Then the required rules were inserted in the knowledge base of this system. Afterward, a value from 0 to 1 was obtained for each sentence in the output based on sentence characteristics and the available rules in the knowledge base. The value obtained in the output determined the degree of the importance of the sentence to be present or absent in the final summary.

In another study by Hannah et al., which used fuzzy logic for summarization, the same procedure was used. Seven features of sentences were extracted [33]. The values for the extracted features were given to fuzzy inference system and fuzzified to identify the important sentences. Based on the importance of a sentence, the fuzzy system defuzzified sentences into one of the three variables named unimportant, average, and important, which were used in selecting sentences for the summary. The summary was generated by the sentences that were ranked important. Provided that the summary size was not satisfactory, sentences with an average rank were used as well. However, unimportant sentences were never used for summary [33].

The results obtained were compared with the Microsoft summarizer to evaluate the performance of this method. In order to perform a comparison task, 55 documents from DUC2002 were selected. The fuzzy system produced an average precision of 0.47, an average recall of 0.49, and an average F-measure of 0.48. Compared with the Microsoft summarizer with the 0.46, 0.39, and 0.42 precision, recall, and F-measure, the fuzzy system had a better performance [33].

Machine learning method is effective in learning the features that are essential to make a summary but there should be a training corpus to learn from it, so the training corpus is different from language to language and is not fixed for every document.

3.3. Semantic-based method

When it comes to summarization, statistical features are not all effective across the board because some features depend on the specific format and the writing style of the documents [34]. For example, in *title word*, the problem is that a document may not have a title. It may also happen that all the frequent words are not as important as lesser frequent words, in which case, ignoring words with less occurrence would provide an unqualified summary. Semantic-based methods identify the relationship between words and sentences by use of a thesaurus, part-of-speech tagging, grammar analysis, and selection of meaningful sentences to generate summary [35]. Various techniques like *lexical chain*, *clustering*, and *graph-based* methods have been developed in this approach, which will be explained in the following sections.

3.3.1. Lexical chain method

The lexical chain is a sequence of words in a document that are semantically related to one another. This method includes three steps, text segmentation, lexical chain identification, and finally, finding the strongest lexical chain for sentence extraction [36]. After segmenting the sentences of a document, the words that are related semantically are identified, and relevant word chains are generated. For identifying a lexical chain, WordNet is used. WordNet is used for determining words that belong to the same synset (synonym set). That is to say, words that occur within the same synset in the WordNet are also semantically related to one another. The semantic relation between words is represented by synonym, hyponym, and meronym [37]. A procedure for constructing lexical chains follows three steps. These steps are as follow:

1. Select a set of candidate words;
2. For each candidate word, find an appropriate chain relying on a relatedness criterion among members of the chains;
3. If it is found, insert the word in the chain and update it accordingly.

At the end, the strong chains are selected towards constructing the final summary. In order to find the strong chain, each chain is scored. The length and homogeneity index are used as two parameters to score the chains. The length is calculated as some words in the chain, and the homogeneity index is 1- the number of distinct words divided by the length. The final score of a chain is computed as the product of these two parameters. The following criterion is used to select the strong chain:

Score (Chain) > Average (Score) + 2 * Standard Deviation (Score)

The chains whose scores exceeded the above criterion are considered as strong chains. Then sentences with the strong chain are selected to form the final summary. In order to evaluate the performance of this method, Nenkova and McKewon [6] compared the results with Microsoft summarizer on the documents that were selected from TREC collection and used the precision and the recall measures. The result precision and recall for lexical chain were 47 and 64, respectively, for 20% of the input text. In comparison with Microsoft summarizer, which had the precision and recall of 32 and 39, the lexical chain had a better performance. It has to be mentioned that a lexical chain considers the relationship between words instead of just focusing on the occurrence of the word. However, it is heavily dependent on the WordNet that is constructed by human [6].

The lexical chain was also utilized by Saxena [38] for text summarization, where the following formula was presented for a stronger lexical chain:

Length (LC) = total number of particular chain members (candidate words) (2)

$$Sig(LC) = \frac{Length(LC)}{\sum_{l \in D} length(l)} * \frac{\log 2 length(LC)}{\sum_{l \in D} length(l)} \quad (3)$$

where, LC is a lexical chain; D, is the document; l, is each chain in document D, and Sig, is the significance of lexical chain in the document.

Related (w, LC) = 1 if they are related (4)
= 0 if they are not related

The above formula is the relation equation that shows if a word w is in the lexical chain LC or not.

$$Utility(LC, D) = sig(LC) * \sum_{\forall w \in D} related(w, LC) \quad (5)$$

The Utility function shows the contribution of the lexical chain in the document.

Score Chain (L) = AVG + 2 * STD. Dev (6)

where, AVG is the average of scores of lexical chain (utility of each chain), and STD. Dev is the standard deviation of the utility scores of each lexical chain.

Lexical chain LC will be considered as a strong chain if Utility (LC) > Score Chain (L).

Evaluation of the results of this method, which is done by the Rouge tool, gives the average precision of 0.45 and the average recall of 0.54.

3.3.2. Clustering method

In a clustering method, similar textual units (such as words, sentences or paragraph) are clustered together to identify the common information between them [39]. Each cluster is known as a sub-domain of the content, with less similarity with other clusters but the components of the cluster have the most likeness with each other [40]. As the number of sentences in the cluster increases, the importance of the information contained within the cluster increases. After grouping the sentences in some clusters, from each cluster, a sentence is selected to produce an extracted summary. These sentences can be chosen using simple positional features.

Sarkar [41] considered four steps for the task of clustering sentences. These steps are pre-processing, sentence clustering, cluster ordering, and selecting representative sentences from clusters. In the pre-processing step, stop word removal, tokenization, and stemming are done. For clustering sentences, the similarity between sentences should be measured to group similar sentences in a cluster; to do this task, a uni-gram matching-based similarity measure is used, which is defined as follows:

$$sim(S_i, S_j) = (2 * |S_i \cap S_j|) / (|S_i| + |S_j|) \quad (7)$$

where, S_i and S_j are two sentences; $|S_i \cap S_j|$ shows the number of matching words between two sentences, and $|S_i|$ is the length of sentence S_i.

After clustering sentences, clusters are ordered, and clusters that have more important words get a high rank. The following equation is used to compute a rank for clusters:

$$weight\ of\ a\ cluster\ C, W(C) = \sum_{w \in C} \log(1 + count(w)) \quad (8)$$

where, count(w) represents the count of the word w. for selection of representative sentences, the author used the method of local and global important word of a sentence. The local importance shows the importance of words in a topic of the cluster but the global importance is the importance of the word on multiple topics. The importance of a sentence is calculated as follows:

$$Score(S) = \sum_{w \in S} Weight(w) \quad (9)$$

$$Weight(w) = \alpha_1 \log(1 + CTF) + \alpha_2 \log(1 + CF)) \quad (10)$$

where, α_1 and α_2 are set to 0.5, Weight (w) shows the importance of the word w, CTF (cluster term frequency) represents the count of a word in the cluster, and log(1+CTF) is used for the local importance of a word. CF (cluster frequency) shows the number of clusters that contain the word, and log(1+CF) is used to calculate the global importance of a word. After ordering clusters and selecting representative sentences, the summary is generated.

For clustering multiple documents, a method has been introduced by Gupta [11], which has 4 steps: pre-processing for preparing the input text, feature extraction to select features based on the output summary feature, single-document summarization whose sentences are clustered in each document, and the final steps of multi-document summarization; a sentence from each cluster is extracted to generate summary. The result of testing this method on the DUC 2002 had the precision of 0.34, recall of 0.33, and F-measure of 0.33.

Another work on text summarization based on clustering has been done by Deshpande and Lobo [42]. The proposed approach is a multi-document system that is also query-based. In their study, they used a collection of different documents and queries as input. Subsequently, similar documents were clustered into one group. For every document, the sentences were then clustered into sentence clusters. For scoring a sentence, several features such as noun feature, cue phrase, sentence length, numerical data, sentence location, sentence centrality, uppercase word, and sentence similarity with the user query are used. In this approach, TF*IDF is used to score sentences. The cosine similarity is used to find the similarity between sentences and queries. Sentences in each document cluster are clustered into sentence clusters based on the similarity value. After scoring each sentence, high scored sentences are selected to form the final summary. The evaluation results show that the proposed method has the values of 0.59, 0.49, and 0.51 for precision, recall and F-score, respectively. Comparing this method with the statistical feature-based method, which yielded 0.49, 0.49, and 0.49, respectively, and the document clustering method with yields of 0.41, 0.3, and 0.32 for precision, recall, and F-score, the proposed method outperforms these two methods.

The clustering method is very useful for multiple-document summarization since different sentences with the same topic in the documents are grouped, and repeated sentences are avoided to be included in the summary. The drawback of the clustering method is that each sentence or paragraph is assigned to only one cluster but some sentences express more than one topic, and should not be restricted to one subject [6]. As it will be discussed in the next section, the graph-based approach does not suffer from this side-effect.

3.3.3. Graph-based method

In the field of natural language processing, graphs are used to display the structure of the text and the connection between sentences. Sentences are represented as a node, and the relation between sentences are depicted by edges [43]. The graph-based text summarization method is a technique to extract a significant, appropriate, and informative text in a compressed version [44]. In order to use this technique, a pre-processing phase should be done on the input text to remove stop words, tokenize the sentences, and so on. Then sentences are ranked to identify the important sentences. Afterward, the relation between sentences is computed to recognize the relevant sentences. At the end, sentences are extracted for the summary based on the ranked and relevant sentences. Kumar et al. [45] used the graph-based technique to summarize Hindi texts. The TF-IDF (Term Frequency-Inverse Document Frequency) method is used to identify the important sentences and rank them. Since the TF-IDF method is used for multi-document, the normalized term is frequency applied for scoring sentences that are defined as follows:

$$TF_{noun} = \alpha + (1-\alpha) * \frac{tf}{tf_{\max}} \tag{11}$$

where, α is between 0 and 1. In order to find the relevant sentences, similar sentences should be identified, so the cosine similarity is used to compute the similarity between two sentences, and is defined as (12):

$$\cos\theta = \frac{\sum_{i=1}^{n} x_i * y_i}{\left(\sqrt{\sum_{i=1}^{n} x_i^2} * \sqrt{\sum_{i=1}^{n} y_i^2}\right)} \tag{12}$$

If two sentences are semantically similar, there should be a connection between them. Extracting sentences for generating a summary is done based on the above two equations; sentences with high ranks and all their relevant sentences are extracted and put together to produce a summary based on the order of input document.

By testing and analyzing this technique on a different document, with a 60% compression rate, the average Recall, Precision, and F-measure were 66.67%, 77.78%, and 70%, respectively.

A recent graph-based text summarization has been done by Natesh *et al.* [46]. In this approach, a pre-processing phase is done on the text including tokenizing, part of speech tagging, and pronoun resolution. After simplifying the text, a graph is built, in which the nodes are the representation of nouns, and the weight of each edge connecting two nodes is used to show the relevance between two nouns. Then the distance between two nouns is computed as follows:

Distance(n1, n2)=|position(n1)-position(n2)| (13)

The edge weight is computed as follows:

Edge_weight(n1,n2)=1/(1+(distance(n1,n2)) (14)

n1 and n2 are nouns.

In order to score the sentences, the following equation is calculated:

$$SentenceScore\,(s) = \forall n \in s \sum relevence\,(n) \quad (15)$$

$$relevence\,(n) = \sum Ni = 0\ edgeWeight\,(n,i) \quad (16)$$

where, n is a noun, N is the total number of nouns, and s is the sentence. After calculating the sentence score, the sentences with a high score are chosen for the summary.

Rouge-1 is used for evaluating the summary; it had the average precision of 0.33, recall of 0.33, and F-score of 0.33.

Semantic-based methods are useful since they consider the meaning of each sentence and word that make a coherent and meaningful summary but using these techniques are time-consuming and require more efforts than the other techniques.

3.4. Swarm intelligence-based method

In a computational context, a swarm is a group of simple agents that have a collective behavior to perform a complex task by acting as a community [47]. The social behavior of ants, termites, bees, and other social beings have motivated researchers to discover their lifestyle. Swarm intelligence is also a branch of artificial intelligence, which is based upon computer simulation to copy the creature's interactions with each other and with their environment to solve an optimization problem [48]. Various algorithms based on the swarm intelligence behavior have been introduced for the problems of optimization [49], robotic [50], routing [51], data mining [52], clustering [53], and so on. In the following sections, some swarm-based algorithms such as particle swarm optimization, cuckoo optimization algorithm, and bacterial foraging optimization along with their usage in text summarization will be explained.

3.4.1. Particle swarm optimization

The particle swarm optimization algorithm is an evolutionary one, which has been introduced for solving an optimization problem [54]. This algorithm is based upon the social movement of birds, and starts with a population of birds to discover the search space. Every individual in a population has a random position and a velocity that is dynamically adjusted not only by the information of its own experience but also by the local knowledge of its neighbors [55]. The best position found by a particle as well as the best position found by all particles are kept in the memory. The next position and velocity of each particle to reach the best value will be updated according to the following equations:

$$V_{id}(t+1) \leftarrow w * V_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) + c_2 r_2 (p_{gd}(t) - x_{id}(t)) \quad (17)$$

p_{id} is the best local solution found by the *i-th* particle, p_{gd} is the best global solution found by all particles, $V_{id}(t)$ is the velocity of a particle at time *t*, r_1 and r_2 are random numbers between [0,1], c_1 and c_2 are acceleration parameters, and *W* is the inertia weight whose value is between [0.4,0.9] [56].

$$x_{id}(t+1) \leftarrow x_{id}(t) + V_{id}(t+1) \quad (18)$$

$x_{id}(t+1)$ is the next position, $x_{id}(t)$ is the current position, and $V_{id}(t+1)$ is the new velocity that is obtained from (2).

Particle swarm optimization has been used to solve various problems. Some of its usages are in clustering [57, 55], scheduling [58], and data mining [59], amongst others.

Particle swarm optimization has also been used in text summarization by Binwahlan [60] for scoring the sentences to generate summary with a high match to human's summary. There are 3 phases in this work: feature selection, PSO encoding, and evaluation function. The first five features are extracted, and in the PSO encoding step, the position of each particle is represented by a bit, corresponding to a feature and has the value 0 or 1. If the value of a bit is 1, it means that the corresponding feature is selected; in the case of 0 value of a bit, the corresponding feature is not selected. Then for each sentence, a score is computed as follows:

$$Score(s_i) = \sum_{j=1}^{5} s(f_j) \times vopp(i) \tag{19}$$

where, $Score(s_i)$ is the score of the sentence s_i, $s(f_i)$ is the score of the *j-th* feature, and vopp(i) is the value of the *i-th* bit in the particle position. The sentences are ranked in the descending order of their scores, and the *n* top sentences are selected for the summary. The resulting summary is evaluated by a fitness function, which shows that pbest is the best-generated summary by a particle and gbest is the best-generated summary by all particles. These two values are used to change the position of each particle. After each iteration, the position of the particle with the gbest value is selected as a vector for the best-selected features. Then the feature weights of all data collections are calculated to compute the final feature weights.

The performance of this algorithm was compared to MS WORD using the Rouge software. The documents of Doc. 2002 including d105g, d070f, d067f, and d061j were used to evaluate the task. The F-Score results for performing the PSO algorithm on the document are 0.42869, 0.44637, 0.40616, and 0.39517, respectively, and the results of MS WORD are 0.41201, 0.36625, 0.38179, and 0.32773. Comparison of the results shows that the PSO algorithm has a better performance [61].

3.4.2. Cuckoo optimization algorithm

Cuckoo optimization is an evolutionary algorithm proposed by Yang and Deb [62] for finding an optimal solution. This algorithm is inspired by the behavior of cuckoo bird in laying eggs. To increase the chances of her egg survival, cuckoo bird lays her eggs in the nest of other birds. Some eggs are discovered by the host bird and are thrown out. Eggs that are similar to the host bird's egg, on the other hand, have the chance to be hatched and grow up. There are three general rules for this algorithm, which are as follow [62]:

- Each cuckoo lays one egg at a time, and leaves its egg in the nest that is chosen randomly;
- The nest with a high quality of eggs is known as the best nest and will be transferred to the next generations;
- The number of available host nests is fixed, and a host can discover an alien egg. In this case, a host bird either throws the egg away or abandons the nest.

The chicks grow up and become mature cuckoo birds. The mature cuckoos make some communities and search the space to find the best environment for breeding and reproduction. Finally, they converge to a location in which a maximum number of cuckoo birds can survive [63]. This location is also known as a global maximum of the objective function. When flying toward the goal habitat, the cuckoos do not fly the entire path. They fly λ% of the destination and with a deviation of φ radians (Figure 2) [64]. λ is a random number between 0 and 1, and φ is a number between $[-\pi/6, \pi/6]$. These two parameters help the cuckoo bird to search for more space.

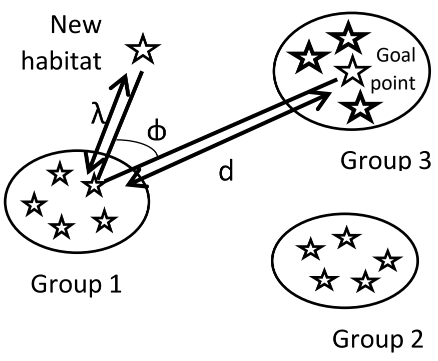

Figure 2. Movement of a cuckoo toward goal habitat [64].

The cuckoo search optimization algorithm was used for text summarization by Mirshojaei and Masoomi [61] to improve the performance of extractive summarization techniques. The proposed method has four steps. The first entails a pre-processing on input document, then a score is assigned to a sentence by the weighting method. Afterward, the similarity between sentences and keyword is obtained using a similarity matrix. Then the cuckoo search optimization algorithm is applied to extract the important sentences, which include five steps [61]:

1. The CSOA parameters are initialized;
2. The sentences are assigned to birds randomly;
3. The assessment of birds is done based on cost function;
4. The position of birds is updated;

5. The end condition of loops is checked, if it is satisfied, the algorithm is finished; otherwise, return to step 3.

The cost function is used to compute the sentence coherence and readability, and is as follows:

$$CF_s = \frac{\log(C * 9 + 1)}{\log(M * 9 + 1)} \tag{20}$$

$$C_s = \frac{\sum_{\forall S_i, S_j \in Summary\ Subgraph} W(s_i, s_j)}{N_s}$$

$$RF_s = \frac{R_s}{\max_{\forall i} R_i} \quad , \quad R_s = \sum_{0 \le i < s} W(s_i, s_{i+1}) \tag{21}$$

where, CF_s is the coherence factor of sentences, C_s is the average similarity of available sentences in summarization, and *M* is the maximum weight of sentences. RF_s is the readability factor of a summary with the length of *s*.

The F-Score result of this algorithm in comparison with PSO algorithm and MS WORD on d105g, d070f, d067f, and d061j are 0.49761, 0.46476, 0.47128 and 0.42391, respectively, which show a better performance in performing the cuckoo search algorithm in text summarization [61].

3.4.3. Bacterial foraging optimization algorithm

The bacterial foraging optimization algorithm is an optimization algorithm, which is inspired by the social behavior of E. Coli bacteria in the body, proposed by Passino [65]. This algorithm consists of 3 steps, which are defined as follow:

- Chemotaxis: this step simulates the movement of bacteria in the state of swim and tumble, and is calculated as follows:
- $$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(l)}}$$
-
- Where, $\theta^i(j,k,l)$ represents the location of i-th bacteria in j-th chemotaxis, k-th reproduction, and l-th elimination-dispersal step, and C(i) is the size of chemotaxis.
- Reproduction: the weak bacteria are destroyed, and the strong bacteria are divided into two bacteria; this helps to balance the population of bacteria.
- Elimination-dispersal: some temporal and sudden situations may kill the bacteria. Some bacteria may also be moved to other places. Some bacteria are killed and some others are initialized randomly to simulate this situation.

This algorithm was used for automatic text summarization by Dastkhosh Nikoo et al. [66]. For doing the task of summarization, the weight is assigned to each word of a sentence by a weighting method. Then by summing over the word's weight, a sentence score is achieved. According to this score, sentences are stored in a descending order. Each word in a sentence is represented by a bit that has the value of 0 or 1. If the value is 1, it means that the term is selected for the summary; in the case of 0 value, the term is not chosen for the summary. After ordering sentences, according to bit mapping, each bacterium selects the word to be in the document summary. The generated summary is evaluated by a fitness function. This process is continued until the bacterial value converges to the threshold value.

By performing this algorithm for the task of summarization on the d105g, d070f, d067f, and d061j documents of the Doc. 2002, the F-measure results were 0.43543, 0.44126, 0.41765 and 0.39121. In comparison with MS WORD, PSO, and cuckoo search algorithm, BFOA had a better performance than MS WORD and PSO but cuckoo search algorithm performed better than BFOA [61].

To give a comprehensive view of different automatic text summarization methods and a comparison of their results after testing on various documents, also the advantages and disadvantages, we collected the results in table 1.

The swarm-based summarization technique gives a valuable summary when the length of the text is long. Performing these methods is done in a shorter time compared to the other summarization methods.

4. Evaluation techniques

In order to evaluate an automatic text summary, it is the content of the summary that ought to be evaluated. Several metrics have been introduced so far to achieve this goal. There are two general categories of evaluation metrics: intrinsic and extrinsic [6]. The main aim of intrinsic metrics is to evaluate the performance of an automated summary content by comparing it with an ideal summary. There are two measures for intrinsic metric. The first measure involves quality evaluation, which tries to check that the summary does not have grammatical errors, does not contain redundant information, and the produced summary possesses structural coherence. The second measure is the content evaluation metric. Content evaluation metrics are divided into two groups: one is co-selection measure and the other is content-based metric.

The co-selection measure shows the number of ideal sentences contained within an automatic summary. The co-selection measure consists of a precision, recall, and F-measure [67]. Precision is computed by the intersection of summarized extracted, and ideal sentences, divided by all extracted sentences. The recall is computed by the intersection between the relevant and retrieved sentences, divided by all the relevant sentences. F-measure is an average of precision and recall criteria, and determines the score of the final set of the selected sentences in a produced summary.

$$precision = \frac{relevant\ senteces \cap retrieved\ sentences}{retrieved\ sentences} \tag{23}$$

$$recall = \frac{relevant\ senteces \cap retrieved\ sentences}{relevant\ sentences} \tag{24}$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recall} \tag{25}$$

Table 1:. Comparison between different methods.

Name of method	List of the studies	Advantages	Disadvantages	Recall	Precision	F-measure
Statistical method	Luhn (word frequency) Baxendale (location method) Edmundson (cue phrase, title words, and sentence location)	Simple to perform	Do not consider the meaning of the word depending on the text format	-	-	-
Machine learning method	Kupiec (naïve Bayes method) Kyoomarsi (fuzzy logic) Kaikhah (artificial neural network)	By utilization of various machine learning algorithms, comprehensive summary is produced	Dependence on the training dataset Complexity in computation	0.47	0.49	0.48
Semantic-based method	Brazilly and Elhadad (lexical chain method)	Consider Word sense disambiguation that has a multiple meaning	Dependence on the WordNet	64	47	-
	Sarkar (sentence clustering) Gupta (multi-document summarization)	Reduces redundancy in multi-document summarization; it is appropriate for the domain-specific summarization	Limits each sentence to be put only in one cluster	0.34	0.33	0.33
	Kumer (graph-based method)	Consider different meanings of sentences	Complexity in measurement	77.78	66.67	70
Swarm intelligence based method	Binwahllan (particle swarm optimization)	Consider each feature of text fairly based on their importance	Stick to the local minima and has early convergence	-	-	0.44
	Mirshojaei et al. (Cuckoo search optimization method)	Better performance compared with other methods	-	-	-	0.46
	Dastkhosh Nikoo et al. (Bacterial foraging optimization method)	-	-	-	-	0.44

Content-based metrics are computed as follow:

- Cosine similarity: by having the two vectors $\vec{z}$ and $\vec{y}$, cosine similarity measures the angle between these two vectors [68, 69].

Supposing that X and Y are the automatic text summary and the reference summary, respectively, the cosine similarity for these two summaries is calculated as follows:

$$\cos(X,Y) = \frac{\sum_i x_i \cdot y_i}{\sqrt{\sum_i (x_i)^2} \cdot \sqrt{\sum_i (y_i)^2}} \quad (26)$$

- Unit overlap [69]: the overlap between two units (vocabulary, phrase or other textual units) of text is calculated as follows:

$$overlap(X,Y) = \frac{\|X \cap Y\|}{\|X\| + \|Y\| - \|X \cap Y\|} \quad (27)$$

where, X and Y are an automatic text summary and a reference summary, respectively.

- ROUGE: Recall-Oriented Understudy for Gisting Evaluation [70] evaluates summary quality by comparing it with a human-generated summary. There are five evaluation metrics in Rouge tool: ROUGE-N, ROUGE-W, ROUGE-S, ROUGE-SU, and ROUGE-L. Rouge-n is computed as follows:

$$Rouge-n = \frac{\sum_{C \in RSS} \sum_{gram_n \in C} Count_{match}(gram_n)}{\sum_{C \in RSS} \sum_{gram_n \in C} Count(gram_n)} \quad (28)$$

where, RSS denotes the reference summary, and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a generated summary and reference summary. Also Count(gram$_n$) is the number of n-grams in the reference summary.

The extrinsic evaluation metric is known as task-oriented measures; question-answering, and information retrieval are instances of the extrinsic method. Their goal is to evaluate a summary performance based on a special task [6].

5. Conclusion

By the progress in the production of a voluminous body of information due to the arrival of the internet, automatic text summarization has attracted a significant level of attention to ease the task of providing necessary information for users. Different kinds of summaries such as abstractive, extractive, and single and multi-documents have been explained in this survey. Also various summarization techniques like statistical, machine learning, semantic-based, and swarm intelligence-based method were described, whose focuses were on the extractive summary since abstractive summary needs complex natural language processing method. Finally, some evaluation methods were introduced, which could be used to examine and compare the results of different approaches. By comparing different methods of summarization, we came into this conclusion that by utilizing different methods in a hybrid way, the quality of summary would be more effective since combining two methods, for example, causes to eliminate their shortcomings and use their strength to improve the quality of the proposed hybrid method. Also the combination of different features of the document often produces better results when assigning weights to sentences.

References

[1] Jezek, K. & Steinberger, J. (2008). Automatic text summarization, The state of the art 2007 and new challenges, in Proceedings of Znalosti, pp. 1-12.

[2] Hovy, E. (2005). text summarization, In: R. Mitkov, Ed., The Oxford Handbook of Computational Linguistics, Oxford University Press, Oxford, pp. 583-598.

[3] Mani, I. (2001). Automatic Summarization. John Benjamins Publishing Company, Philadelphia, Pennsylvania, U.S.A, pp. 1-286.

[4] Jones, K. S. (1999). Automatic summarization: factors and directions. Advance in automatic text summariztion, Springer, pp. 1-12.

[5] Hovy, E. & Lin, C. Y. (1999). Automated Text Summarization in SUMMARIST. In I. Mani and M.T.Maybury, eds., Advances in Automatic Text Summarization, The MIT Press, pp.81–94.

[6] Nenkova, A. & McKeown, K. (2011). Automatic summarization. Foundations and Trends in Information Retrieval. vol. 5, no. 2-3, pp. 103-233.

[7] Kumar, Y. J., Goh, O. S., Basiron, H., Choon, N. H. & Suppiah, P. C. (2016). A Review on Automatic Text Summarization Approaches. Journal of Computer Science, vol. 4, no. 12, pp. 178-190.

[8] Jade, G., Mittal, V., Carbonell, J. & Kantrowitz, M. (2000). Multi-document summarization by sentence exraction. In proceeding of the 2000 NAACL-ANLPWorkshop on automatic summarization. Seattle, Washington, pp. 40-48.

[9] Saggion, H. & Lapalme, G. (2002). Generating Indicative-Informative Summaries with SumUM. Computation of linguistic, vol. 28, no. 4, pp. 497-526.

[10] Fan, W., Wallace, L. & Zhongj, S. R. (2005). Tapping into the Power of Text Mining. Communications of ACM, Vol. 49, No. 9, pp. 76-82.

[11] Gupta, V. & Lehal, S. L. (2010). A Survey of Text Summarization Extractive Techniques. Journal of emerging technologies in web intelligence, vol. 2, no.3, pp. 258-268.

[12] Cheung, J. (2008). Computing abstractive and extractive summarization of evalutive text:

controversality and content selection. B. Sc. (Hons.) University of British Columbia, pp. 1-38.

[13] Karmakar, S., Lad, T. & Chothani, H. (2015). A Review Paper on Extractive Techniques of Text Summarization. International Research Journal of Computer Science (IRJCS), vol. 2, no. 1, pp. 18-21.

[14] Gong, Y. & Liu, X. (2011). Generic Text Summarization using Relevance Measure and Latent Semantic Analysis. In Proceedings of the Annual. International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, USA, pp. 19-25.

[15] Kumar, C., Pingali, P. & Varma, V. (2008). Generating Personalized Summaries Using Publicly Available Web Documents. in ACM International Conference on Web Intelligence and Intelligent Agent Technology, IEEE Computer Society Washington, DC, USA, vol. 3, pp. 103-106.

[16] Wu, C. W. & Liu, C. L. (2003). Ontology-based Text Summarization for Business News Articles. in Proceedings of the ISCA Eighteenth International Conference on Computers and Their Applications, pp. 389-392.

[17] Radev, D. R., Fan, W. & Zhang, Z (2001). WebInEssence: A Personalized Web-Based Multi-Document Summarization and Recommendation System. NAACL Workshop on. Automatic Summarization, pp. 79-88.

[18] Reeve, L. H., Han, H. & Brooks, A. D. (2007). The use of domain-specific concepts in biomedical text summarization. Information Processing & Management, vol. 43, no. 6, pp. 1765–1776.

[19] Hassel, M. & Mazdak, N. (2004). FarsiSum - A Persian text summarizer. In the proceedings of Computational Approaches to Arabic Script-based Languages, Workshop at Coling 2004, the 20th International Conference on Computational Linguistics, pp. 82-84

[20] Hovy, E. & Lin, C. Y. (1999). Automated Text Summarization in SUMMARIST. In Advances in Automatic Text Summarization, I. Mani and M. Maybury (editors), pp. 1-14.

[21] Luhn, H. (1958). The Automatic Creation of literature abstracts. IBM J. Res. Dev, pp. 159-165.

[22] Baxendale, P. B. (1958). Machine- Made Index for Technical Literature. An Experiment. IBM Journal of Research Development, vol. 4, no. 2, pp. 354-361.

[23] Edmondson, H. P. (1969). New Methods in Automatic Extracting. Journal of the ACM, vol. 2, no. 16, pp. 264-285.

[24] Oak, R. (2016). Extractive Techniques for Automatic Document Summarization: A Survey. International Journal of Innovative Research in Computer and Communication Engineering, vol. 4, no. 3, pp. 4158-4164.

[25] Othman, B., Haggag, M. & Belal, M. (2014). A taxonomy for text summarization. Information Science and Technology, vol. 3, no. 1, pp. 43-50.

[26] Bharti, S. K., Babu, K. S. & Jena, S. K. (2017). Automatic keyword extraction for text summarization in multi document e-newspapers article. European Journal of Advances in Engineering and Technology, vol. 4, no. 6, pp. 410-427.

[27] Gambhir, M. & Gupta, V. (2016). Recent automatic text summarization techniques: a survey. Artif. Intell. Rev. Springer Sci. Media Dordr., vol. 47, no. 1, pp. 1–66.

[28] Kupiec, J., Pedersen, J. & Chen, F. (1995). A trainable document summarizer. In SIGIR '95 Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, Seattle, Washington, USA, pp. 68-73.

[29] Kaikhah, K. (2004). Text Summarization Using Neural Networks. In proceeding of second conference on intelligent system, IEEE, vol. 1, pp. 40-44.

[30] Zadeh, L. (1965). Fuzzy Sets. information and control, vol.8, no.3, pp. 338-353.

[31] Zadeh, L. (1999). From Computing with Numbers to Computing with Words—From Manipulation of Measurements to Manipulation of Perceptions. IEEE transaction on circuits and systems: fundamental theory and application, vol. 45, no. 1, pp. 105-119.

[32] Kyoomarsi, F., Khosravi, H., Eslami, E., Dehkordy, P. K. & Tajoddin, A. (2008). Optimizing Text Summarization Based on Fuzzy Logic. in Seventh IEEE/ACIS International Conference on Computer and Information Science, pp. 347-352.

[33] Hannah, M. E., Geetha, T. & Mukhe, S. (2011). Automatic Extractive Text Summarization Based on Fuzzy Logic: A Sentence Oriented Approach. Springer-Lecture Notes in Computer Science, pp. 530-538.

[34] Youngjoong, K. & Jungyun, S. (2008). An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. Pattern Recognition Letters, vol. 29, no. 9, pp. 1366–1371.

[35] Chandra, M., Gupta, V. & Paul, S. K. (2011). A Statistical approach for Automatic Text Summarization by Extraction. In 2011 International Conference on Communication Systems and Network Technologies, pp. 268-271.

[36] Barzilay, R. & Elhadad, M. (1997). Using Lexical Chain for Text Summarization. In Proceeding of the ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization, pp. 10-17.

[37] Ranjan, P. A. & Saha, D. (2014). An approach to automatic text summarization using WordNet. In

Advance Computing Conference (IACC), pp. 1169-1173.

[38] Saxena, S. & Saxena, A. (2016). An Efficient Method based on Lexical Chains for Automatic Text Summarization. International Journal of Computer Applications, vol. 144, no. 1, pp. 47-52.

[39] Hatzivassiloglou, V., Klavans, J. L., Holcombe, M. L., Barzilay, R., Kan, M. Y. & McKeown, K. R. (2001). SIMFINDER: A flexible clustering tool for summarization. In proceeding of the NAACL Workshop on Automatic Summarization, pp. 41-49.

[40] Khazaei, A. & Ghasemzadeh, M. (2015). Comparing k-means clusters on parallel Persian-English corpus. Journal of AI and Data Mining, vol. 3, no. 2, pp. 203-208.

[41] Sarkar, K. (2009). Sentence Clustering-based Summarization of Multiple Text Documents. TECHNIA – International Journal of Computing Science and Communication Technologies, vol. 2, no. 1, pp. 325-335.

[42] Anjali, D. R. & Lobo, L. M. R. J. (2013). Text Summarization using Clustering Technique. International Journal of Engineering Trends and Technology (IJETT), vol. 4, no. 8, pp. 3348-3351.

[43] Ramesh, A., Srinivasa, K. G. & Pramod, N. (2014). SentenceRank- A graph based approach to summarize text. In Application of Information and Web Technologies (ICADIWT), Bangalore, India, pp. 177-182.

[44] Mihalcea, R. & Tarau, P. (2005). A Language Independent Algorithm for Single and Multiple Document Summarization. In Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP), pp. 602-607.

[45] Kumar, K. V., Yadav, D. & Sharma, A. (2015). Graph Based Technique for Hindi Text Summarization. Information Systems Design and Intelligent Applications, vol. 339, pp. 301-310.

[46] Natesh, A. A., Balekuttira, S. T. & Patil, A. P. (2016). Graph Based Approach for Automatic Text Summarization. International Journal of Advanced Research in Computer and Communication Engineering, vol. 5, no. 2, pp. 6-9.

[47] Jarraya, B. & Bouri, A. (2012). Metaheuristic Optimization Backgrounds: A Literature Review. International Journal of Contemporary Business Studies, vol. 3, no. 12, pp. 31-44.

[48] Ajith, A., Swagatam, D. & Sandip, R. (2008). Swarm Intelligence Algorithms for Data clustering. in Soft Computing for Knowledge Discovery and Data Mining, O. Maimon and L. Rokach, Eds., Springer US, pp. 279-313.

[49] Bonabeau, E., Dorigo, M. & Theraulaz, G. (1999). Swarm Intelligence: From Natural to Artificial Swarm intelligence, Oxford University Press, pp. 1-320.

[50] Beni, G. (2005). From Swarm Intelligence to Swarm Robotics. International Workshop on Swarm Robotics, Springer, Berlin, Heidelberg, pp. 1-9.

[51] Saleem, M., Di Caro, G. A. & Farooq, M. (2011). Swarm intelligence based routing protocol for wireless sensor networks: Survey and future directions. Information Sciences, vol. 181, no. 20, p. 4597–4624.

[52] Grosan, C., Abraham, A. & Chis, M. (2006). Swarm Intelligence in Data Mining, In Swarm Intelligence in Data Mining, Springer Berlin Heidelberg, vol. 34, pp. 1-20.

[53] Ajith, A., Swagatam, D. & Sandip, R. (2008). Swarm Intelligence Algorithms for Data Clustering. in Soft Computing for Knowledge Discovery and Data Mining, Springer US, pp. 279-313.

[54] Kennedy, J. & Eberhart, R. (1995). Particle Swarm Optimization, In proceeding of the IEEE international joint conference on neural networks, vol. 4, pp. 1942-1948.

[55] Izakian, Z. & Mesgari, S. (2015). Fuzzy clustering of time series data: A particle swarm optimization approach, Journal of AI and Data Mining, vol. 3, no. 1, pp. 39-46.

[56] Eberhart, R. & Shi, Y. (2001). Particle swarm optimization: Developments, applications and resources. in proceedings of the 2001 Congress on Evolutionary Computation, Seoul, pp. 81-86.

[57] Cura, T. (2012). A particle swarm optimization approach to clustering. Expert Systems with Applications: An International Journal, vol. 39, no. 1, pp. 1582-1588.

[58] Pandey, S., Wu, L. & Guru, S. M. (2010). A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. The 24th IEEE Int. Conf. on Advanced Information Networking and Applications, pp. 400-407.

[59] Sousa, T., Silva, A. & Neves, A. (2004). Particle Swarm based Data Mining Algorithms for classification tasks. Parallel Computing, vol. 30, no. 5-6, pp. 767–783.

[60] Binwahlan, M. S., Salim, S. & Suanmali, S. (2009). Swarm Based Features Selection for Text Summarization. IJCSNS International Journal of Computer Science and Network Security, vol. 9, no. 1, pp. 175-179.

[61] Mirshojaei, S. H. & Masoomi, B. (2015). Text Summarization Using Cuckoo Search Optimization Algorithm. Journal of Computer & Robotics, vol. 8, no. 2, pp. 19-24.

[62] Yang, X. S. & Deb, S. (2009). Cuckoo Search via L´evy Flights. in proceeding of World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), pp. 210-214.

[63] Lashkari, M. & Moattar, M. (2017). Improved COA with Chaotic Initialization and Intelligent Migration for Data Clustering. Journal of AI and Data Mining, vol. 5, no. 2, pp. 293-305.

[64] Rajabioun, R. (2011). Cuckoo Optimization Algorithm, Applied Soft Computing, vol. 11, no. 8, pp. 5508–5518.

[65] Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control, IEEE control system magazine, vol. 22, no. 3, pp. 52-67.

[66] Nikoo, M. D., Faraahi, A., Hashemi, S. M. & Erfani, S. H. (2012). A Method for Text Summarization by Bacterial Foraging Optimization Algorithm, IJCSI International Journal of Computer Science Issues, vol. 9, no. 4, pp. 36-40.

[67] Steinberger, J. & Jezek, K. (2009). Evaluation measures for text summarization. In Computing and Informatics, vol. 28, no. 2, pp. 1001-1028.

[68] Salton, G. (1988). Automatic text processing. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, pp. 1-523.

[69] Saggion, H., Radev, D., Teufel, S., Lam, W. & Strassel, S. M. (2002). Developing Infrastructure for the Evaluation of Single and Multi-Document Summarization Systems in a Cross-Lingual Environment. In Proceedings of LREC, Las Palmas, pp. 747-754.

[70] Lin, C (2004). Rouge: A package for automatic evaluation of summaries. In Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, vol. 8, pp. 74-81.4

نشریه هوش مصنوعی و داده کاوی

مقاله مروری خلاصه‌سازی متن

نسرین نظری و محمدامین مهدوی*

دانشکده فنی مهندسی، دانشگاه بین المللی امام خمینی (ره)، قزوین، ایران.

ارسال ۲۰۱۷/۰۸/۲۰؛ بازنگری ۲۰۱۷/۱۱/۰۴؛ پذیرش ۲۰۱۸/۰۲/۰۵

چکیده:

خلاصه‌سازی متن به معنی تولید نسخه‌ای کوتاه‌تر از متن می‌باشد که محتوای اصلی متن در این فرایند حفظ می‌گردد. تولید محتوای متنی به خصوص در صفحات وب روز به روز درحال افزایش می‌باشد. استخراج اطلاعات مفید از بین حجم عظیمی از اسناد به یک چالش اساسی تبدیل شده است که نیازمند خودکار نمودن این فرایند می‌باشد. سیستم‌های خلاصه‌سازی متن تلاش می‌نمایند که با حفظ اطلاعات مرتبط و حذف بخش‌های غیرمرتبط متن به این فرایند کمک نماید. بر اساس ورودی، یک سیستم خلاصه‌سازی متن دارای دو رویکرد می‌باشد. در رویکرد اول، که خلاصه‌سازی تک سندی نامیده می‌شود، خلاصه‌ای از یک سند تولید می‌شود. به عبارت دیگر، سیستم یک سند را به عنوان ورودی می‌پذیرد و یک خلاصه تولید می‌نماید. در رویکرد دوم، سیستم چندین سند که موضوع آنها یکسان می‌باشد را به عنوان ورودی می‌پذیرد و یک خلاصه واحد تولید می‌نماید که به عنوان خلاصه-سازی چند سندی در نظر گرفته می‌شود. براساس خروجی نیز، سیستم‌های خلاصه‌سازی به دو دسته تقسیم می‌شوند. دسته اول که خلاصه‌های استخراجی می‌باشند، که با استخراج جملات متن اصلی و کنار هم قرار دادن آنها به تولید خلاصه می‌پردازد. دسته دوم با بررسی و تحلیل متن به تولید خلاصه‌ای می‌پردازد که با جملات و کلماتی متفاوت از متن اصلی تولید می‌شود. در این مقاله، قصد داریم برخی از سیستم‌های خلاصه‌سازی خودکار متن را مرور نماییم. همچنین برخی از شاخص‌های ارزیابی را به منظور ارزیابی کیفیت خلاصه‌های تولید شده توسط سیستم‌های خلاصه‌ساز بررسی می‌نماییم.

کلمات کلیدی: خلاصه‌سازی خودکار متن، خلاصه‌سازی چندسندی، خلاصه‌سازی تک سندی، شاخص‌های ارزیابی خلاصه.