Expert Systems with Applications 165 (2021) 113679

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Review

Automatic text summarization: A comprehensive survey

Wafaa S. El-Kassas [a,*], Cherif R. Salama [a,b], Ahmed A. Rafea [b], Hoda K. Mohamed [a]

[a] Department of Computer and Systems Engineering, Faculty of Engineering, Ain Shams University, Cairo, Egypt
[b] Department of Computer Science and Engineering, The American University in Cairo, Cairo, Egypt

ARTICLE INFO

Article history:
Received 11 October 2019
Revised 26 April 2020
Accepted 18 June 2020
Available online 11 July 2020

Keywords:
Automatic text summarization
Text summarization approaches
Text summarization techniques
Text summarization evaluation

ABSTRACT

Automatic Text Summarization (ATS) is becoming much more important because of the huge amount of textual content that grows exponentially on the Internet and the various archives of news articles, scientific papers, legal documents, etc. Manual text summarization consumes a lot of time, effort, cost, and even becomes impractical with the gigantic amount of textual content. Researchers have been trying to improve ATS techniques since the 1950s. ATS approaches are either extractive, abstractive, or hybrid. The extractive approach selects the most important sentences in the input document(s) then concatenates them to form the summary. The abstractive approach represents the input document(s) in an intermediate representation then generates the summary with sentences that are different than the original sentences. The hybrid approach combines both the extractive and abstractive approaches. Despite all the proposed methods, the generated summaries are still far away from the human-generated summaries. Most researches focus on the extractive approach. It is required to focus more on the abstractive and hybrid approaches. This research provides a comprehensive survey for the researchers by presenting the different aspects of ATS: approaches, methods, building blocks, techniques, datasets, evaluation methods, and future research directions.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

The web resources on the Internet (e.g. websites, user reviews, news, blogs, social media networks, etc.) are gigantic sources of textual data. Besides, there is a wealth of textual content on the various archives of news articles, novels, books, legal documents, biomedical documents, scientific papers, etc. The textual content on the Internet and other archives grow exponentially on a daily basis. As a result, users consume a lot of time to find the information they are looking for. They cannot even read and comprehend all the textual content of search results. There are many repeated or unimportant portions of the resulting texts. Therefore, summarizing and condensing the text resources becomes urgent and much more important. Manual summarization is an expensive task and consumes a lot of time and effort. Practically, it is very difficult for humans to manually summarize this huge amount of textual data (Vilca & Cabezudo, 2017). The Automatic Text Summarization (ATS) is the key solution to this dilemma.

The main objective of an ATS system is to produce a summary that includes the main ideas in the input document in less space (Radev, Hovy, & McKeown, 2002) and to keep repetition to a minimum (Moratanch & Chitrakala, 2017). The ATS systems help the users to get the main points of the original document without the need to read the entire document (Nazari & Mahdavi, 2019). The users will benefit from the automatically produced summaries and they will save a lot of time and effort. In Maybury (1995), Maybury defined the automated summary as follows: "An effective summary distills the most important information from a source (or sources) to produce an abridged version of the original information for a particular user(s) and task(s)". In Radev et al. (2002), Radev et al. also defined the summary as follows: "A summary can be loosely defined as a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that. Text here is used rather loosely and can refer to speech, multimedia documents, hypertext, etc.". The produced summary should be shorter in length than the input text and include the most important information in the input text (Gambhir & Gupta, 2017).

ATS systems can be classified as single-document or multi-document summarization systems. The former produces the summary from a single document while the latter generates the summary from a cluster of documents. ATS systems are designed by applying one of the text summarization approaches: extractive, abstractive, or hybrid. The extractive approach selects the most

* Corresponding author.
E-mail addresses: wafaa.elkassas@gmail.com (W.S. El-Kassas), cherif.salama@eng.asu.edu.eg (C.R. Salama), rafea@aucegypt.edu (A.A. Rafea), hoda.korashy@eng.asu.edu.eg (H.K. Mohamed).

https://doi.org/10.1016/j.eswa.2020.113679
0957-4174/© 2020 Elsevier Ltd. All rights reserved.

important sentences from the input text and uses them to generate the summary. The abstractive approach represents the input text in an intermediate form then generates the summary with words and sentences that differ from the original text sentences. The hybrid approach combines both the extractive and abstractive approaches. The different classifications for ATS systems are defined in Section 2. The general architecture of an ATS system; as shown in Fig. 1; consists of the following tasks:

1. Pre-Processing: producing a structured representation of the original text (Gupta & Lehal, 2010) using many linguistic techniques like sentences segmentation, words tokenization, removal of stop-words, part-of-speech tagging, stemming, etc.
2. Processing: using one of the text summarization approaches by applying a technique or more to convert the input document(s) to the summary. Section 3 defines the different ATS approaches and Section 4 explores the different techniques and building blocks to implement an ATS system.
3. Post-Processing: solving some problems in the generated summary sentences like anaphora resolution and reordering the selected sentences before generating the final summary.

ATS is one of the most challenging tasks in Natural Language Processing (NLP) and Artificial Intelligence (AI) in general. ATS research began as early as 1958 with Luhn's work (Luhn, 1958) that automatically excerpts abstracts of magazine articles and technical papers. ATS poses many challenges to the research community like: 1) identification of the most informative segments in the input text to be included in the generated summary (Radev et al., 2002), 2) summarization of long single documents such as books, 3) summarization of multi-documents (Hahn & Mani, 2000), 4) evaluation of the computer-generated summary without the need for the human-produced summary to be compared with, and 5) generation of an abstractive summary (Hahn & Mani, 2000) similar to a human-produced summary. Researchers still dream of an accurate ATS system to produce a summary that 1) covers all the main topics in the input text, 2) does not include redundant or repeated data, and 3) is readable and cohesive to the users. Since the beginning of ATS research in the late 1950s, they have been trying and are still working to improve techniques and methods for generating summaries so that the computer-generated summaries can match with the human-made summaries (Gambhir & Gupta, 2017).

Many surveys have been recently published about ATS systems and methodologies. Most surveys focus on techniques and methods of extractive summarization like Nazari and Mahdavi (2019) because the abstractive summarization needs extensive NLP. In Kirmani, Manzoor Hakak, Mohd, and Mohd (2019), Kirmani et al. define the common statistical features and some extractive methods. In Kumar and Sharma (2019), Kumar and Sharma provide a survey about the extractive ATS systems that apply fuzzy logic methods. In Mosa, Anwar, and Hamouda (2019), Mosa et al. provide a survey on how the swarm intelligence optimization techniques are applied for ATS. They aim to motivate researchers to use swarm intelligence optimization for ATS especially for short text summarization. Suleiman and Awajan (2019) provide a survey about extractive deep-learning-based text summarization. Some surveys focus on abstractive summarization like Gupta and Gupta (2019), Lin and Ng (2019) for different abstractive methods and Tandel, Mistree, and Shah (2019) for abstractive neural-network-based methods. Some surveys focus on a domain-specific summarization like (Bhattacharya et al., 2019; Kanapala, Pal, & Pamula, 2019) for legal documents summarization, (Dutta et al., 2019) for comparing extractive algorithms used in the micro-blogs summarization, and Jacquenet, Bernard, and Largeron (2019) for abstractive deep-learning-based methods and challenges of meeting summarization. Some surveys like (Gupta, Bansal, & Sharma, 2019; Mahajani, Pandya, Maria, & Sharma, 2019) provide a review about some abstractive and extractive methods. A survey about the summarization evaluation methods is presented in (Ermakova, Cossu, & Mothe, 2019).

Most of the state-of-the-art surveys tackle a subset of the ATS aspects like focusing on one approach (e.g. extractive summarization), one method for a specific approach (e.g. extractive fuzzy logic method), one specific-domain ATS systems (e.g. legal documents summarization), etc. Besides, Dutta et al. (2019) highlights that different ATS algorithms produce different summaries from the same input texts so it is very promising to combine outputs from multiple ATS algorithms to produce better summaries. Also, Mahajani et al. (2019) conclude that it is recommended to benefit from the advantages of both extractive and abstractive approaches by proposing hybrid ATS systems. Therefore, the main motivation for this work is to provide a comprehensive survey about the various ATS aspects in order to help researchers enhance computer-generated summaries potentially by combining different approaches and/or methods. The main contributions of this research include:

- Illustrating the classifications of the ATS systems and explaining the different ATS applications provided with examples of ATS systems proposed in the literature for each application.
- Providing a systematic review about the ATS approaches: extractive, abstractive, and hybrid. Each approach is applied using several methods in the literature. This survey provides: 1) the general architecture, advantages, and disadvantages of each approach and 2) the methods of each approach along with the advantages, disadvantages, and examples of ATS systems for each method. A conclusion about the state-of-the-art research recommendations for each approach is provided at the end of the approach subsection in Section3.
- Providing an overview about the various components and techniques that are used to design and implement the ATS systems. The survey illustrates: 1) the text summarization operations inspired from the analysis of the human experts' operations, 2) the widely-used statistical and linguistic features that identify the important words and sentences, and 3) the text summarization building blocks. The building blocks include: 1) the common text representation models, 2) the linguistic analysis

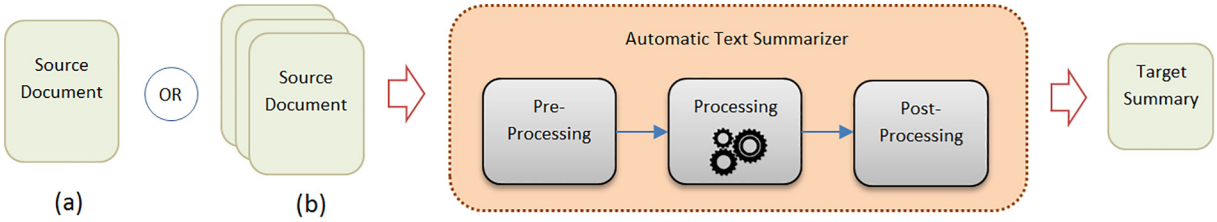

Fig. 1. (a) Single-document or (b) Multi-document, automatic text summarizer.

and processing techniques, and 3) the soft computing techniques (e.g. machine learning, optimization algorithms, and fuzzy logic).
- Providing an overview about the standard datasets besides the manual evaluation criteria and automatic evaluation tools for the computer-generated summaries.
- Providing a listing and categorization of the future research directions for the research community based on the existing ATS systems limitations and challenges.

The structure of this paper mirrors its main contributions: Section 2 illustrates the various classifications of ATS systems and the ATS applications. Section 3 introduces ATS approaches and explores the different methods proposed in the literature for each approach. Section 4 highlights the techniques and building blocks that are used to implement ATS systems, while Section 5 explores the benchmarking datasets and ATS evaluation methods. Finally, Section 6 concludes the paper and provides future directions for ATS research.

2. Ats systems classifications and applications

This section explores the different ways to classify ATS systems and highlights their different applications.

2.1. Classifications of the ATS systems

There are many classifications for ATS systems as illustrated in Fig. 2. ATS systems can be classified based on any of the criteria below.

Classification based on the Input Size: Single-document or Multi-document. Input size refers to the number of source documents used to generate the target summary. As shown in Fig. 1, Single-Document Summarization (SDS) uses a single text document to generate a summary and the target is to shorten the input document while keeping the important information (Joshi, Wang, & McClean, 2018). In Multi-Document Summarization (MDS), the summary is generated based on a set of input documents and the target is to remove repetitive content in the input documents (Joshi et al., 2018). MDS is more complex than SDS and has some prominent issues like redundancy, coverage, temporal relatedness, compression ratio, etc. (Gupta & Siddiqui, 2012).

Classification based on the Text Summarization Approach: Extractive, Abstractive, or Hybrid. The extractive text summarization approach selects the most important sentences in the input document(s) and the output summary is composed by concatenating the selected sentences. The abstractive text summarization approach represents the input document(s) in an intermediate representation and the output summary is generated from this representation. Unlike extractive summaries, abstractive summaries consist of sentences that are different than the original document (s) sentences. The hybrid text summarization approach merges between both the extractive and abstractive approaches. These approaches will be explained in more detail in section 3.

Classification based on Nature of the Output Summary: Generic or Query-Based. A generic text summarizer extracts important information from one or more input documents to provide a general sense of its contents (Gambhir & Gupta, 2017; Sahoo, Balabantaray, Phukon, & Saikia, 2016). A query-based summarization means that the multi-document summarizer deals with a group of homogeneous documents, taken out from a large corpus as a result of a query (Sahoo et al., 2016). Then, the generated summary includes a query-related content. A query-based summary presents the information that is most relevant to the initial search query, while a generic summary gives an overall sense of the document content (Mohan, Sunitha, Ganesh, & Jaya, 2016). The query-based summary is sometimes referred to as a like query-focused, topic-focused, or user-focused summary (Gambhir & Gupta, 2017).

Classification based on the Summary Language: Monolingual, Multilingual, or Cross-Lingual. A summarization system is monolingual when the language of source and target documents is the same. A summarization system is multi-lingual when the source text is written in a number of languages (e.g. English, Arabic, and French) and the summary is also generated in these languages. A summarization system is cross-lingual when the source text is written in one language (e.g. English) and the summary is generated in another one (e.g. Arabic or French) (Gambhir & Gupta, 2017).

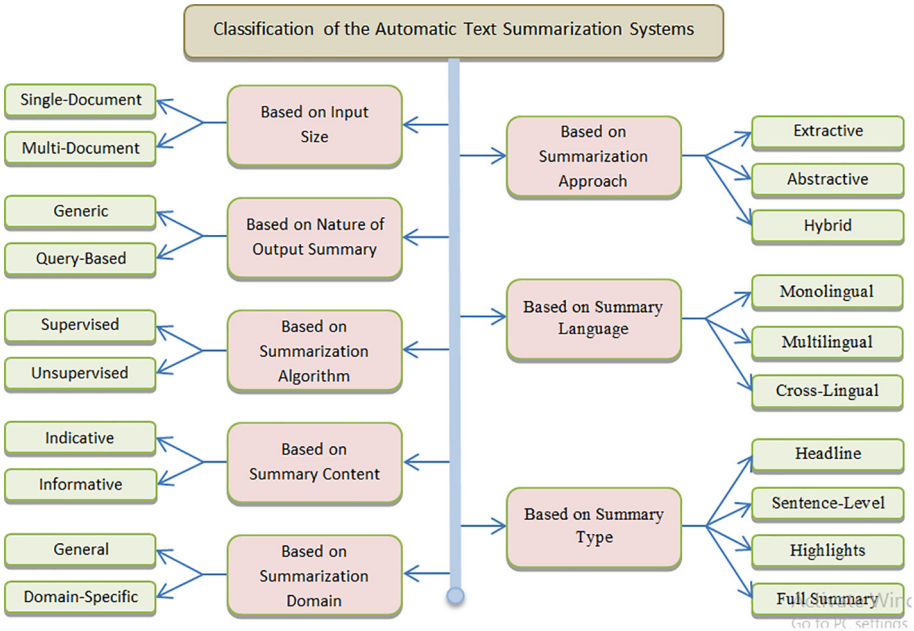

Fig. 2. Classifications of the ATS systems.

Classification based on the Summarization Algorithm: Supervised, or Unsupervised. The supervised algorithm needs a training phase which requires an annotated training data. Human efforts are required to manually annotate the training data, so the latter is difficult to create and expensive. On the other side, the unsupervised algorithm does not require a training phase nor training data (Mohd, Jan, & Shah, 2020).

Classification based on the Summary Content: Indicative or Informative. An indicative summary contains only the general idea or information about the source text (Bhat, Mohd, & Hashmy, 2018). So it is used to determine what the input text is about (i.e. what topics are addressed) to alert the user about the source content (Takeuchi, 2002). The purpose of an indicative summary is to inform users about the scope of the input text to help them decide whether or not to read the original text. On the other side, an informative summary contains the important information and ideas in the original text (Bhat et al., 2018) such that it covers all topics of the text (Gupta & Lehal, 2010). The purpose of an informative summary is to cover the main contents of the original text without details (Takeuchi, 2002).

Classification based on the Summary Type: Headline, Sentence-Level, Highlights, or Full Summary. The length of the generated summaries differs based on the purpose of the ATS system. Headline generation produces a headline which is usually shorter than a sentence (Dernoncourt, Ghassemi, & Chang, 2018). A sentence-level summarization generates a single sentence from the input text which is usually an abstractive sentence (Dernoncourt et al., 2018). A highlights summarization produces a telegraphic style and highly compressed summary which is usually in the form of bullet points (Woodsend & Lapata, 2010). The highlights summary provides the reader with a brief overview about the main information in the input document(s) (Woodsend & Lapata, 2010). Finally, a full summary generation is usually guided by the required summary length or a compression ratio.

Classification based on the Summarization Domain: General, or Domain-Specific. The general or domain-independent ATS system summarizes documents that belong to different domains. On the other side, the domain-specific ATS system is specialized to summarize documents from a certain domain (e.g. medical documents or legal documents).

2.2. Applications of the ATS systems

ATS is widely used in text mining and analytics applications such as information retrieval, information extraction, question answering, etc. ATS is used with the information retrieval techniques to enhance the capabilities of the search engines. In Tuarob, Bhatia, Mitra, and Giles (2016), Tuarob et al. propose a search engine to search for algorithms and pseudo-codes. First, a dataset is constructed by extracting algorithms from scientific papers. Then, ATS is used to add additional textual metadata to the extracted algorithms from the scientific documents. In Yulianti, Chen, Scholer, Croft, and Sanderson (2018), Yulianti et al. use text summarization to extract answers for non-factoid queries. In Li, Zhu, Ma, Zhang, and Zong (2018), Li et al. propose an extractive multi-modal summarization (MMS) system for asynchronous collections of text, image, audio, and video. The proposed system generates a textual summary from the aforementioned sources.

There are different text genres like news articles, novels, books, forums, blogs, emails, opinion reviews, spoken dialogues which combine text summarization with speech recognition, medical documents, legal documents, etc. (Vodolazova, Lloret, Muñoz, & Palomar, 2013a). Each ATS system supports one or more text genres as inputs; hence the ATS systems are used for different applications like news summarization, email summarization, domain-specific summarization (e.g. legal or biomedical documents summarization), etc. Examples of various ATS applications are explored below.

News Summarization: Newsblaster (McKeown et al., 2002) is a text summarizer that helps users find the most desirable news to them. On a daily basis, the system automatically collects, clusters, categorizes, and summarizes news from several news sites on the Internet (e.g. CNN and Reuters) (Gupta & Lehal, 2010). Sahni and Palwe (2017), Sethi, Sonawane, Khanwalker, and Keskar (2017) are other examples of ATS systems for news summarization.

Opinion/Sentiment Summarization: Sentiment Analysis (SA) is the study of people's opinions, emotions, and judgments about events and products. ASFuL is an aspect-based sentiment summarization system (Mary & Arockiam, 2017). It uses fuzzy logic to classify sentiments or opinions polarity from the product reviews as "Strong Positive", "Positive", "Negative", and "Strong Negative". It also integrates the non-opinionated sentences using the Imputation of Missing Sentiment (IMS) technique. IMS is used to reduce the neutral score in sentiment polarity. E-commerce websites are growing rapidly hence there are hundreds of reviews for any product on average. ATS is very useful in this case to summarize the user reviews. Bhargava and Sharma (2017), Lovinger, Valova, and Clough (2019), Mirani and Sasi (2017), Roul and Arora (2019), Yadav and Chatterjee (2016), Zhou, Wan, and Xiao (2016) are additional examples of ATS systems for opinion/sentiment or user reviews summarization.

Microblog/Tweet Summarization: Social networking sites like Facebook, Twitter, etc. include millions of messages (Vijay Kumar & Janga Reddy, 2019). During emergency events, microblogging sites such as "Twitter" are important sources of real-time information because hundreds, thousands or even more of microblogs (tweets) are posted on Twitter (Dutta et al., 2019). Therefore, microblog summarization became very important in recent years. Examples of ATS systems for microblog or tweet summarization can be found in Chakraborty, Bhavsar, Dandapat, and Chandra (2019), Rudra, Goyal, Ganguly, Imran, and Mitra (2019), Vijay Kumar and Janga Reddy (2019).

Books Summarization: most research focus on short documents summarization. In Mihalcea and Ceylan (2007), Mihalcea and Ceylan address the problems of book summarization and introduce a specific benchmark for book summarization. ATS systems developed for short documents are not suitable for summarizing long documents such as books because: 1) the selected sentences may not cover all the book topics, 2) considering the length of documents is essential for better performance, and 3) using the sentence position feature differs in books than short documents (i.e. it may be unuseful to include the sentences located at the beginning of the book in the generated summary).

Story/Novel Summarization: In Kazantseva and Szpakowicz (2010), Kazantseva and Szpakowicz present an approach for automatic extractive summarization that uses syntactic information and shallow semantics (provided by a GATE gazetteer (Cunningham, Maynard, Bontcheva, & Tablan, 2002)) to produce indicative summaries focusing on three types of entities: people, locations, and timestamps of short stories. The generated summary helps the reader to decide whether to read the complete story, but it does not attempt to retell the plot for two reasons: 1) many people do not want to know what happens in a story before reading it, and 2) the summarization problem would be too complex if summarizing the full plot was required.

Email Summarization: Email messages are domain-general text, they are unstructured and not always syntactically well-formed (Muresan, Tzoukermann, & Klavans, 2001). In Muresan et al. (2001), Muresan et al. propose an ATS system that combines linguistic techniques with machine learning algorithms to extract noun phrases to generate a summary of Email messages. More

examples of ATS systems for email summarization can be found in Carenini, Ng, and Zhou (2007), Ulrich, Carenini, Murray, and Ng (2009).

Biomedical Documents Summarization: In Menéndez, Plaza, and Camacho (2014), Menéndez et al. propose an ATS system that combines genetic clustering and graph connectivity information to improve the graph-based summarization process. Genetic clustering identifies the different topics in a document, and connectivity information (i.e. degree centrality) shows the importance of the different topics. Morales, Esteban, and Gerv (2008), Nasr Azadani, Ghadiri, and Davoodijam (2018), Reeve, Han, and Brooks (2006), Reeve, Han, and Brooks (2007) are other examples of ATS systems for biomedical text summarization.

Legal Documents Summarization: In Kavila, Puli, Prasada Raju, and Bandaru (2013), Kavila et al. propose an ATS system and an automatic search system for legal documents to save the time of legal experts. The summarization task identifies the rhetorical roles presenting the sentences of a legal judgment document. The search task identifies the related past cases based on the given legal query. The hybrid system uses different techniques such as keyword or key phrase matching technique and the case-based technique. Anand and Wagh (2019), Kavila et al. (2013), Merchant and Pande (2018) are among examples of ATS systems for legal documents summarization.

Scientific Papers Summarization: Scientific papers are well-structured documents that have some common characteristics like the predictable locations of typical items in a document, cue words, and a template-like structure (Kazantseva & Szpakowicz, 2010). In Mohammad et al. (2009), Mohammad et al. propose a multi-document ATS framework that generates a technical survey on a given topic from multiple research papers by merging two methods: 1) mining the structure of citations and relations among citations to get citation information, and 2) summarization techniques that identify the content of the material in both the citing and cited papers. (Alampalli Ramu, Bandarupalli, Nekkanti, & Ramesh, 2020) propose a summarizer to extract the problem statement from one research paper then uses it to find the related papers. Jiang et al. (2019) present an abstractive deep-learning-based summarizer used for automatic survey generation. Cohan and Goharian (2018), Lloret, Romá-Ferri, and Palomar (2011, 2013), Marques, Cozman, and Santos (2019), Mohammad et al. (2009), Teufel and Moens (2002), Zhang, Li, and Yao (2018) represent various examples of ATS systems for scientific papers.

3. Ats approaches

There are three main text summarization approaches: extractive, abstractive, or hybrid. Each approach is applied using different methods as shown in Fig. 3. This section will provide a detailed overview about each of these approaches along with the methods of each approach in the literature. There are many summarization methods like graph-based, semantic-based, soft computing (SC) based, etc.

3.1. Extractive text summarization

3.1.1. Approach

Fig. 4 shows the extractive text summarization system architecture that consists of 1) pre-processing of the input text, 2) post-processing like: reordering the extracted sentences, replacing pronouns with their antecedents, replacing the relative temporal expression with actual dates, etc. (Gupta & Lehal, 2010), and 3) the processing tasks as follows:

1. Creating a suitable representation of the input text to facilitate the text analysis (e.g. N-gram, bag-of-words, graphs, etc.) (Joshi et al., 2018).
2. Scoring of sentences: ranking sentences based on the input text representation (Nenkova & McKeown, 2012).
3. Extraction of high-scored sentences: selecting the most important sentences from the input document(s) and concatenating them to create the summary (Nenkova & McKeown, 2012; Zhu et al., 2017). The generated summary length depends on the preferred compression rate using a length cutoff or threshold to limit the size of the summary and preserve the same order of the generated sentences as the input text (Wang, Zhao, Li, Ge, & Tang, 2017).

Advantages: The extractive approach is faster and simpler than the abstractive approach. This approach leads to a higher accuracy

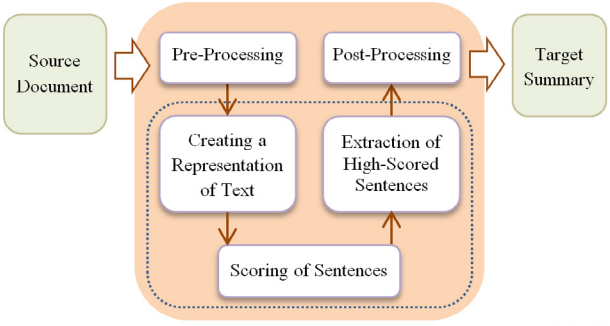

Fig. 4. The architecture of an extractive text summarization system.

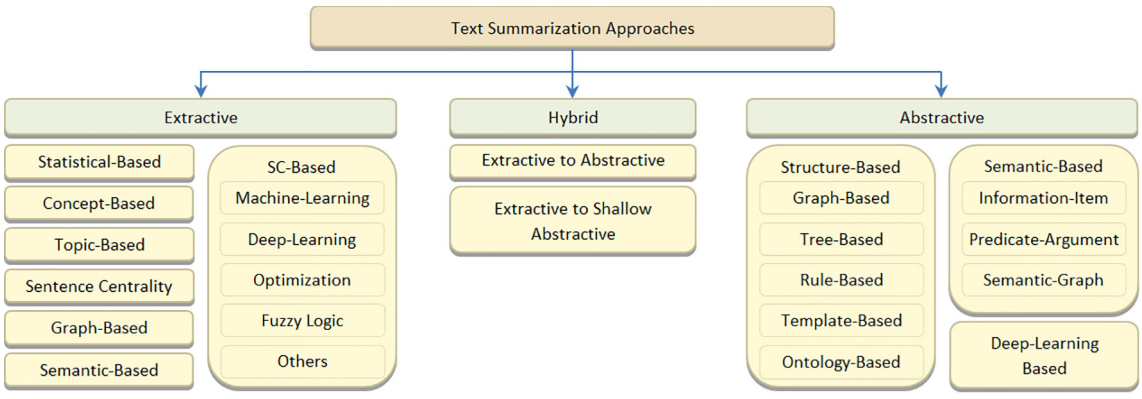

Fig. 3. Automatic text summarization approaches and their associated methods.

because of the direct extraction of sentences so readers read the summary with the exact terminologies that exist in the original text (Tandel, Modi, Gupta, Wagle, & Khedkar, 2016).

Disadvantages: The extractive approach is far from the method that human experts write summaries (Hou, Hu, & Bei, 2017). Drawbacks of the generated extractive summary include:

1. Redundancy in some summary sentences (Hou et al., 2017).
2. Extracted sentences can be longer than average (Gupta & Lehal, 2010).
3. Temporal expressions conflicts in the multi-document case because extractive summaries are selected from different input documents (Gupta & Lehal, 2010).
4. Lack of semantics and cohesion in summary sentences (Moratanch & Chitrakala, 2017) because of the incorrect link between sentences and non-resolved co-reference relationships (Lloret, Romá-Ferri, & Palomar, 2011) and "dangling" anaphora (Gupta & Lehal, 2010).
5. Important information spread across sentences. Conflicting information may not be covered (Gupta & Lehal, 2010). The output summary may be unfair for input texts that consist of several topics (Moratanch & Chitrakala, 2017). Summaries can overcome this issue only if the summary is long enough.

3.1.2. Methods

Fig. 3 shows various extractive text summarization methods like statistical-based, concept-based, etc. Table 1 illustrates the pros and cons of each method along with examples of ATS systems that apply it. In the rest of this subsection, we list and briefly describe each of these methods.

Statistical-Based Methods: These methods extract important sentences and words from the source text based on the statistical analysis of a set of features. The "most important" sentence is defined as the "most favorably positioned", the "most frequent", etc. (Gupta & Lehal, 2010). The sentence scoring steps of a statistical-based extractive summarizer include (Gambhir & Gupta, 2017): 1) selecting and calculating some statistical and/or linguistic features then assigning weights to them (Gupta & Lehal, 2010), and 2) assigning a final score to each sentence in the document (Gambhir & Gupta, 2017) that is determined using a feature-weight equation (Gupta & Lehal, 2010) (i.e. all the selected features' scores are computed and summed to obtain the score of each sentence).

Concept-Based Methods: These methods extract concepts from a piece of text from external knowledge bases like WordNet (Miller, 1995; WordNet: An Electronic Lexical Database, 1998), HowNet, Wikipedia, etc. The importance of sentences is then calculated based on the concepts retrieved from the external knowledge base HowNet instead of words. The sentence scoring steps of a concept-based extractive summarizer include (Moratanch & Chitrakala, 2017): 1) retrieving concepts of a text from the external knowledge base, 2) building a conceptual vector or graph model to show the relationship between concept and sentences, and 3) applying a ranking algorithm to score the sentences.

Topic-Based Methods: These methods rely on identifying a document's topic which is its main subject (i.e. what the document is about). Some of the most common methods for topic representations are Term Frequency, Term Frequency-Inverse Document Frequency (TF-IDF), lexical chains, topic word approaches in which the topic representation consists of a simple table and their corresponding weights (Nenkova & McKeown, 2012), etc. The processing steps of a topic-based extractive summarizer include (Nenkova & McKeown, 2012): 1) converting the input text to an intermediate representation that captures the topics discussed in the input text, and 2) assigning an importance score to each of the sentences in the input document according to this representation.

Sentence Centrality or Clustering-Based Methods: In these methods, a multi-document extractive summarizer identifies the most central and important sentences in a cluster such that they cover the important information related to the cluster main subject. The sentence centrality is defined using the centrality of its words. The common way to measure the word centrality is to look at the centroid of the document cluster in a vector space. The centroid of a cluster is a pseudo-document that consists of words having TF-IDF scores above a predefined threshold (Erkan & Radev, 2004). The sentence scoring steps in a centroid-based summarizer include: 1) building a representative centroid by computing the TF-IDF representations of each sentence in the document (Mehta & Majumder, 2018), and 2) considering a sentence as central when it contains more words from the centroid of the cluster as a measure of its closeness to the cluster centroid (Erkan & Radev, 2004). The closer a sentence is to the cluster centroid the more important it is (Mehta & Majumder, 2018). Clustering-based summarization considers both relevance and redundancy removal in the generated summary. The clustering algorithms are used for ATS and the sentence selection steps include (Nazari & Mahdavi, 2019): 1) using a clustering algorithm to cluster the input sentences, 2) ranking and ordering clusters such that a cluster gets a high rank when it has more important words, and 3) from these clusters, selecting representative sentences for the summary.

Graph-Based Methods: These methods use sentence-based graphs to represent a document or document cluster. Using such representation has been commonly used for extractive summarization (e.g. LexRank (Erkan & Radev, 2004) and TextRank (Mihalcea & Tarau)). In Erkan and Radev (2004), Erkan and Radev discuss how random walks on sentence-based graphs can help in text summarization. For example, LexRank's sentence scoring steps include (Wang et al., 2017): 1) representing the document sentences using an undirected graph such that each node in the graph represents a sentence from the input text and for each pair of sentences the weight of the connecting edge is the semantic similarity between the two corresponding sentences using cosine similarity, and 2) using a ranking algorithm to determine the importance of each sentence. The sentences are ranked based on their LexRank scores in a similar way to the PageRank (Brin & Page, 1998) algorithm except that the LexRank graph is undirected (Moratanch & Chitrakala, 2017).

Semantic-Based Methods: Latent Semantic Analysis (LSA) is a commonly used semantic-based extractive ATS method. LSA is an unsupervised technique that represents text semantics based on the observed co-occurrence of words (Nenkova & McKeown, 2012). The sentence scoring steps of any LSA-based extractive summarizer include (Al-Sabahi, Zhang, Long, & Alwesabi, 2018): 1) creating the input matrix (term-to-sentence matrix), and 2) applying Singular Value Decomposition (SVD) to the input matrix to identify the relationships between terms and sentences. Many other semantic-based techniques are used for ATS like Semantic Role Labelling (SRL) and Explicit Semantic Analysis (ESA). In Mohamed and Oussalah (2019), Mohamed and Oussalah propose a semantic-based extractive ATS system based on the SRL and ESA along with the Wikipedia knowledge base.

Machine-Learning-Based Methods: These methods convert the summarization problem to a supervised classification problem at the sentence level. The system learns by examples to classify each sentence of the test document either as "summary" or "non-summary" class using a training set of documents (i.e. a collection of documents and their respective human-generated summaries). For the machine-learning-based summarizer, the sentence scoring steps include (Moratanch & Chitrakala, 2017): 1) extracting features from the preprocessed document (i.e. based

Table 1
Pros and cons of the extractive text summarization methods.

Method	Pros	Cons	Examples
Statistical-Based	It requires less processor capacity and memory (Gambhir & Gupta, 2017). It does not require any extra linguistic knowledge or complex linguistic processing and it is language independent (Ko & Seo, 2008).	Some important sentences may be not included in the summary because they have less score than others. Similar sentences may be included in the summary because they have high scores.	(Afsharizadeh, Ebrahimpour-Komleh, & Bagheri, 2018)
Concept-Based	The summary sentences cover several concepts.	It requires to use similarity measures in order to reduce redundancy (Moratanch & Chitrakala, 2017). The selected concepts (i.e. they are based on the knowledge base) affect the quality of the summary.	(Sankarasubramaniam, Ramanathan, & Ghosh, 2014)
Topic-Based	The summary sentences concentrate on the various topics in the input document(s).	Sentences that do not have the highest score will not appear in the summary even if they are very related to the main topics (Wang & Ma, 2013). The selected topics affect the quality of the generated summary.	(Sahni & Palwe, 2017)
Sentence Centrality or Clustering-Based	It may avoid including repeated sentences in the summary. It is suitable for multi-document summarization because it groups different sentences about the same topic in the documents (Nazari & Mahdavi, 2019).	It requires prior specification of the number of clusters. The highly scored sentences may be similar, therefore redundancy removal techniques are required (Mehta & Majumder, 2018). Some sentences may express more than one topic but each sentence has to be assigned to only one cluster (Nazari & Mahdavi, 2019).	(Radev, Jing, Styś, & Tam, 2004; Roul & Arora, 2019)
Graph-Based	It enhances coherency and detects redundant information (Moratanch & Chitrakala, 2017). It is language-independent (Nasar, Jaffry, & Malik, 2019), and domain-independent (Nallapati, Zhai, & Zhou, 2017).	It assumes that the weights of the words are equal, so it does not consider the importance of words in the document (Fang, Mu, Deng, & Wu, 2017). It does not focus on issues like the dangling anaphora problem (Moratanch & Chitrakala, 2017). Graphs that represent sentences as Bag of Words and use similarity measure; may fail to identify semantically equivalent sentences (Khan et al., 2018). The selected sentences are affected by the accuracy of similarity computation (Wang et al., 2017).	(Baralis, Cagliero, Mahoto, & Fiori, 2013; Dutta, Das, Mallick, Sarkar, & Das, 2019; Mallick, Das, Dutta, Das, & Sarkar, 2019; Mihalcea, 2004; Sarracén & Rosso, 2018)
Semantic-Based	LSA is a language-independent technique and generates semantically related sentences in the summary (Gambhir & Gupta, 2017).	The generated summary depends on the quality of the semantic representation of the input text. Computing SVD consumes a large time (Gambhir & Gupta, 2017).	(Mashechkin, Petrovskiy, Popov, & Tsarev, 2011; Mohamed & Oussalah, 2019)
Machine-Learning-Based	To improve the sentence selection for the summary, a large set of training data is required (Moratanch & Chitrakala, 2017). Relatively-simple regression models can achieve better results than other classifiers (Gambhir & Gupta, 2017).	It requires a large data set of manually created extractive summaries such that each sentence in the original training documents can be labeled as either "summary" or "non-summary" (Moratanch & Chitrakala, 2017).	(Alguliyev et al., 2019; John & Wilscy, 2013; Shetty & Kallimani, 2017)
Deep-Learning-Based	The network can be trained based on the human reader's style. The features set can be changed based on the user's requirements (Moratanch & Chitrakala, 2017).	It requires human efforts to manually build the training data. Neural networks are slow in the training and testing phases. It is hard to define how the network generates a decision (Moratanch & Chitrakala, 2017).	(Cheng & Lapata, 2016; Kobayashi et al., 2015; Nallapati et al., 2017; Warule, Sawarkar, & Gulati, 2019; Yao, Zhang, Luo, & Wu, 2018; Yousefi-Azar & Hamey, 2017)
Optimization-Based	Using the strength of genetic algorithms to find the optimal weights (Meena & Gopalani, 2015).	High computational time and cost. It is required to define the number of iterations for the optimization algorithms.	(Lovinger et al., 2019; Sanchez-Gomez, Vega-Rodríguez, & Pérez, 2020a; Verma & Om, 2019)
Fuzzy-Logic-Based	Fuzzy logic is compatible with the real world which is not a two-value (0 and 1) world (Nazari & Mahdavi, 2019). It handles the uncertainties in the input as the fuzzy inference systems can produce a logical assessment in an uncertain and ambiguous environment (Kumar & Sharma, 2019).	The redundancy of the selected sentences in the summary is a negative factor that may occur and affect the quality of summary (Patel et al., 2019). So, a redundancy removal technique is required in the post-processing phase to improve the quality of the final summary.	(Mutlu, Sezer, & Akcayol, 2019; Patel et al., 2019; Qassem, Wang, Barada, Al-Rubaie, & Almoosa, 2019)

on multiple features of sentences and words), and 2) feeding the extracted features to a neural network which produces a single value as an output score.

Deep-Learning-Based Methods: In Kobayashi, Noguchi, and Yatsuka (2015), Kobayashi et al. propose a summarization system using document level similarity based on embeddings (i.e. distributed representations of words). An embedding of a word represents its meaning. A document is considered as a bag-of-sentences and a sentence as a bag-of-words. The task is formalized as the problem of maximizing a sub-modular function defined by the negative summation of the nearest neighbors distances on embedding distributions (i.e. a set of word embeddings in a document). Kobayashi et al. conclude that the document-level similarity can determine more complex meanings than sentence-level similarity. Chen and Nguyen (2019) propose an ATS system for single-document summarization using a reinforcement learning algorithm and a Recurrent Neural Network (RNN) sequence model of encoder-extractor network architecture. The important features

are selected by a sentence-level selective encoding technique then the summary sentences are extracted.

Optimization-Based Methods: These methods convert the summarization problem to an optimization problem. For example, a generic extractive multi-document ATS system is formulated as a multi-objective problem in Sanchez-Gomez, Vega-Rodríguez, and Pérez (2020b). The sentence scoring steps of an optimization-based extractive summarizer include: 1) creating a suitable representation to the input text such as the commonly-used vector representation (i.e. each sentence in the input text is represented as a vector of words), and 2) using an optimization algorithm (e.g. A Multi-Objective Artificial Bee Colony (MOABC) algorithm) to select the summary sentences based on the required summary length limit besides one or more optimization criteria: content coverage, redundancy reduction, relevance and coherence (Sanchez-Gomez et al., 2020b). In addition, the strength of genetic algorithms in adjusting weights could be used for ATS. In Meena and Gopalani (2015), the sentence scoring steps for an extractive genetic-algorithm-based summarizer include: 1) identifying the text features from the input text like sentence location, sentence length, etc. and, 2) using the genetic algorithm to adjust weights of these features then calculating the sentences scores.

Fuzzy-Logic-Based Methods: These methods use the fuzzy logic concept for ATS. Fuzzy logic resembles the human reasoning system and provides an efficient way to represent the feature values of sentences because not everything in the world can be defined as zero and one (Kumar & Sharma, 2019). The sentence scoring steps include (Nazari & Mahdavi, 2019): 1) selecting a set of features for each sentence like sentence length, term weight, etc., and 2) using the fuzzy logic system (i.e. after inserting the required rules in the knowledge base of this system) to provide a score for each sentence that reflects the sentence importance. So, each sentence in the output gets a score value from 0 to 1 based on the sentence features and the predefined rules in the knowledge base.

In conclusion, using different methods together produce better summaries because this combination uses their strength and eliminates their shortcomings. Besides, using a combination of different features most probably produces better results when calculating the weights of the input sentences (Nazari & Mahdavi, 2019). Many ATS systems combine multiple methods to benefit from the advantages of each method like Alami, Meknassi, and En-nahnahi (2019), Mao, Yang, Huang, Liu, and Li (2019), Mohd et al. (2020), Moratanch and Chitrakala (2017), Rahman, Rafiq, Saha, Rafian, and Arif (2019). In Moratanch and Chitrakala (2017), Moratanch and Chitrakala combine both graph-based and concept-based methods to build a summarization system. In Rahman et al. (2019), Rahman et al. propose an extractive ATS system to summarize Bengali text using TextRank, Fuzzy C-Means, and aggregate sentence scoring methods. Mohd et al. (2020) propose an extractive summarizer which uses a Distributional Semantic Model to capture the semantics of text, the K-means clustering algorithm to cluster the semantically similar sentences, and a ranking algorithm to rank sentences in each cluster. Mohd et al. conclude that capturing semantics to be used for summarization improves the precision of the generated summaries. Alami et al. (2019) propose an extractive ATS system relying on an ensemble model with a majority voting technique that combines two models: Bag-of-Words and Sentence2Vec (i.e. it represents each sentence in the input text as a vector). Alami et al. conclude that the ensemble model usually achieves more precise results than a single model because the data of each vector is complementary to each other. Mao et al. (2019) use three different methods of combining unsupervised learning with supervised learning to produce single documents summaries.

3.2. Abstractive text summarization

3.2.1. Approach

Abstractive summarization needs a deeper analysis of the input text (Mohan et al., 2016). Abstractive text summarizers generate a summary by understanding the main concepts in the input document using NLP methods, then paraphrasing the text to express those concepts in fewer words and using a clear language (Al-Abdallah & Al-Taani, 2017; Krishnakumari & Sivasankar, 2018). This approach does not copy sentences from the original text for the summary generation (Bhat et al., 2018) instead it requires the ability to make new sentences. Fig. 5 shows architecture of an abstractive text summarizer. It consists of the pre-processing, post-processing, and the processing tasks which include: 1) building an internal semantic representation (Chitrakala, Moratanch, Ramya, Revanth Raaj, & Divya, 2018), and 2) generating a summary using natural language generation techniques to create a summary that is closer to the human-generated summaries (Chitrakala et al., 2018).

Advantages: It generates better summaries with different words that do not belong to the original text by using more flexible expressions based on paraphrasing, compression, or fusion (Hou et al., 2017). The generated summary is closer to the manual summary (Sun, Wang, Ren, & Ji, 2016). Abstractive methods can further reduce the text when compared to the extractive ones (Wang et al., 2017). This higher condensation is because the production of new sentences can further reduce any redundancy eventually achieving a good compression rate (Sakhare, Kumar, & Janmeda, 2018).

Disadvantages: In practice, generating a high-quality abstractive summary is very difficult (Hou et al., 2017). Good abstractive summarizers are very hard to develop as they require the use of natural language generation technology which itself is still a growing field (Wang et al., 2017). The abstractive approach needs a full interpretation of the input text in order to generate new sentences. Most abstractive summarizers suffer from the generation of repeated words and are not able to deal with out-of-vocabulary words appropriately (Hou et al., 2017). Capabilities of the abstractive summarizers are constrained by the richness of their representations. Systems cannot summarize what their representations cannot capture. In limited domains, it may be feasible to devise appropriate structures, but a general-purpose solution depends on an open-domain semantic analysis (Gupta & Lehal, 2010).

3.2.2. Methods

The abstractive ATS methods can be categorized into three categories (Gupta & Gupta, 2019): 1) structure-based: using predefined structures (e.g. graphs, trees, rules, templates, and ontologies), 2) semantic-based: using the text semantic representation and the natural language generation systems (e.g. based on information items, predicate arguments, and semantic graphs), and 3) deep-learning-based methods. Lin and Ng (2019) provide another categorization to the abstractive methods as neural-based or classical which broadly refers to any method that is not neural-based.

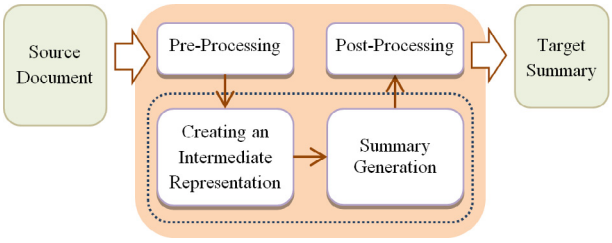

Fig. 5. The architecture of an abstractive text summarization system.

Structure-based methods identify the most important data in the input text then use graphs, trees, rules, templates, or ontology to produce the abstractive summaries (Gupta & Gupta, 2019). Semantic-based methods build the semantic representation of the input text by using the information-items, predicate-arguments, or semantic-graphs then use a natural language generation system to produce the abstractive summaries (Gupta & Gupta, 2019). Various abstractive text summarization methods are shown in Fig. 3 categorized as structure-based, semantic-based, and deep-learning-based. Table 2 illustrates the pros and cons of each method along with examples of summarizers that applying them. We use the rest of this subsection to list and briefly describe each of these methods.

Graph-Based Methods: In Ganesan, Zhai, and Han (2010), Ganesan et al. propose an abstractive summarizer "Opinosis" that uses a graph model. Each node represents a word and positional information is linked to nodes. Directed edges represent the structure of sentences. The processing steps of the graph-based method in Ganesan et al. (2010) include: 1) graph generation: constructing a textual graph to represent the source text, and 2) summary generation: generating the target abstractive summary. To achieve this, various sub-paths in the graph are explored and scored as follows:

1. Rank all the paths then sort their scores in descending order. The ranking also includes the collapsed paths.
2. Eliminate duplicated (or extremely similar) paths by using a similarity measure (e.g. Jaccard).
3. Select the top few remaining paths as the generated summary, with the number of paths to be chosen controlled by a parameter, which represents summary size.

Tree-Based Methods: These methods identify similar sentences that share mutual information then accumulate these sentences to produce the abstractive summary (Gupta et al., 2019). The similar sentences are represented into a tree-like structure. Parsers are used to construct the dependency trees which are the most used tree-form representations for the text. To create the final summaries, some tasks are performed to process the trees like pruning, linearization (i.e. converting trees to strings), etc. (Gupta & Gupta, 2019). Kurisinkel, Zhang, and Varma (2017) propose a multi-document abstractive summarizer as follows: 1) parse the input texts in the corpus to build a set of all syntactic dependency trees, 2) extract a set of partial dependency trees (with variable sizes) from the syntactic dependency trees, 3) cluster the extracted partial dependency trees to guarantee topical diversity, and 4) use the partial trees in each cluster to form a single sentence that represents the cluster in the abstractive summary.

Rule-Based Methods: These methods require defining the rules and categories to discover the important concepts in the input text then use these concepts to produce the summary. The steps of using this method include: 1) categorize the input text based on

Table 2
Pros and cons of the abstractive text summarization methods.

Method	Pros	Cons	Examples
Graph-Based	It is applicable to any domain and it does not require any intervention by human experts (Khan, Salim, & Farman, 2016; Ranjitha & Kallimani, 2017). A new sentence is created by linking all words in a word graph path (Le & Le, 2013).	Word graphs do not reflect the word/phrase meaning. Diverse words/phrases that refer to the same subject are represented as different nodes, so sentences consisting of these nodes cannot be merged (Le & Le, 2013).	(Khan et al., 2016; Lloret et al., 2011; Ranjitha & Kallimani, 2017)
Tree-Based	The generated summaries have improved quality when using language generators because they produce less-redundant and fluent summaries (Gupta & Gupta, 2019).	It cannot identify the relations between sentences without discovering the shared phrases between these sentences. It does not consider the context so it misses many significant phrases in the text. This method performance depends on the available parsers which limit its efficiency. It focuses on the syntax more than the semantics (Gupta & Gupta, 2019).	(Kurisinke, Zhang, & Varma, 2017)
Rule-Based	The generated summaries have high information density (Gupta & Gupta, 2019). The ability to expand the complexity and variety of the abstraction schemes and generation patterns to handle more aspects (Genest & Lapalme, 2012).	Preparing the rules is a time-wasting and tedious process because the rules are hand-crafted and written manually (Gupta & Gupta, 2019).	(Genest & Lapalme, 2012)
Template-Based	It produces informative and coherent summaries and can be used for multi-document summarization as the template slots filled by the snippets extracted by the information extraction rules (Gupta & Gupta, 2019).	The extraction rules and linguistic patterns for template slots need to be created manually. Similar information across multiple documents cannot be handled. (Khan et al., 2016). The templates are predefined so they lack diversity. Similarity and differences among the documents cannot be considered (Gupta & Gupta, 2019).	(Embar, Deshpande, Vaishnavi, Jain, & Kallimani, 2013; Oya, Mehdad, Carenini, & Ng, 2014)
Ontology-Based	It focuses on documents that are related to a specific domain. It provides coherent summaries (Vodolazova & Lloret, 2019) and can handle uncertainties in the text easily (Gupta & Gupta, 2019).	It needs a good ontology but preparing this ontology is a very time-consuming process (Gupta & Gupta, 2019) because it heavily requires domain experts to build the domain ontology. As such it needs more time and effort. It cannot be generalized to other domains (Khan et al., 2016).	(Mohan et al., 2016; Okumura & Miura, 2015)
Semantic-Based	Using the SRL will help providing the semantic relationship between the phrase of words (Ranjitha & Kallimani, 2017).	The generated summary depends on the quality of the semantic representation of the input text.	(Khan et al., 2015)
Deep-Learning-Based	Seq2seq performs very well in short text summarization (Wang et al., 2017).	It requires a huge amount of structured training data. Training of RNN-based Seq2Seq models is slow and they cannot capture distant dependency relations for long sequences (Cai, Shen, Peng, Jiang, & Dai, 2019). Seq2Seq suffers from generating repetitive content and inaccurate information when it is used for noisy social media text (Miao, Zhang, Bai, & Cai, 2019).	(Cai et al., 2019; Chopra, Auli, & Rush, 2016; Hou et al., 2017; Jiang et al., 2019; See, Liu, & Manning, 2017)

the terms and concepts existing in it, 2) formulate the questions based on the domain of the input text, 3) answer the questions by finding the terms and concepts in the text, and 4) fed the answers into some patterns to produce the abstractive summary. For example, questions may be like "what is the event?", "who did the event?", "when the event has occurred?", "where the event has occurred?", "what was the impact of the event?", etc. (Gupta & Gupta, 2019). In Genest and Lapalme (2012), Genest and Lapalme propose an architecture based on the abstraction schemes. Each abstraction scheme is designed to address a subcategory or theme that consists of content selection heuristics, Information Extraction (IE) rules, and straightforward generation patterns are designed for each scheme. All these rules are created manually. An abstraction scheme targets to answer one or more aspects and more than one scheme that can be related to the same aspect. The IE rules can detect several candidates for each aspect and the content selection module can select the best ones to send to the generation module.

Template-Based Methods: For some domains (e.g. meeting summaries), human summaries have shared sentence structures that can be defined as templates. Based on the input text genre, the abstractive summary can be produced by using the data in the input text to fill the slots in the suitable pre-defined templates (Lin & Ng, 2019). Extraction rules and linguistic patterns are used to determine the text snippets which fill the template slots (Gupta et al., 2019).

Ontology-Based Methods: Many documents are related to specific domains and every domain has its own information structure that can be represented by a knowledge dictionary like an ontology (Moratanch & Chitrakala, 2016). The basic idea is to get the proper information from the input text to form an abstractive summary by using an ontology (Okumura & Miura, 2015).

Semantic-Based Methods: These methods represent the input document(s) by a semantic representation (e.g. information items, predicate-argument structures, or semantic graphs) then fed this representation to the natural language generation system which uses the verb and noun phrases to produce the final abstractive summary (Gupta & Gupta, 2019). In Khan, Salim, and Jaya Kumar (2015), Khan et al. propose a multi-document abstractive summarizer that: 1) represents the input documents with predicate-argument structures using SRL, 2) clusters the semantically similar predicate-argument structures across the text using a semantic similarity measure, 3) ranks the predicate-argument structures based on features weighted and optimized using a Genetic Algorithm, and 4) uses the language generation to generate sentences from these predicate-argument structures.

Deep-Learning-Based Methods: The recent success of sequence-to-sequence learning (seq2seq) makes abstractive summarization feasible (Hou et al., 2017). Seq2seq has achieved great success in various NLP tasks like machine translation, voice recognition, and dialogue systems (Wang et al., 2017). A set of RNN models based on attention encoder-decoder achieves promising results for short text summarization; however, deep learning methods still suffer from some problems such as: 1) generating repeated words or phrases, and 2) inability to deal with out-of-vocabulary (OOV) words (i.e. rare and limited-occurrence words) (Hou et al., 2017). The steps of the summarization system in Hou et al. (2017) include: 1) converting the dataset into plain texts and saving the original documents (e.g. news articles) and their summaries separately, 2) applying word segmentation then using a subword model to process the data, 3) initializing the word vectors using the pre-trained Gensim toolkit (Rehurek & Sojka, 2010) that will be further trained in the proposed model, 4) using Tensorflow (Mart et al., 2016) for implementation with one layer of bidirectional Long Short-Term Memory (LSTM) for the encoder and a unidirectional LSTM layer for the decoder. Cross-entropy is used to calculate the loss and the Adam optimizer is used to optimize the loss.

In conclusion, the recent abstractive summarization efforts mainly focus to use the deep learning models especially in short text summarization (Kouris, Alexandridis, & Stafylopatis, 2019). It is recommended to combine different methods and techniques to benefit from their advantages for generating better abstractive summaries. The different ATS algorithms produce different summaries from the same input texts so it is very promising to combine outputs from multiple ATS algorithms to produce better summaries than the summaries generated by using individual algorithms (Dutta et al., 2019). The structure-based methods are usually used with the extractive methods for producing hybrid summaries and with the semantic-based or deep-learning-based methods to produce abstractive summaries (Gupta & Gupta, 2019). For example, these methods can be used in the pre-processing phase to extract the important key-phrases in the input text then use other methods to generate the abstractive summary (Gupta & Gupta, 2019). In Kouris et al. (2019), Kouris et al. propose an abstractive ATS system that combines the semantic-based data transformations and the encoder-decoder deep learning models.

3.3. Hybrid text summarization

3.3.1. Approach

The hybrid approach combines both the abstractive and extractive approaches. The typical architecture of a hybrid text summarizer is shown in Fig. 6. It commonly consists of the following phases (Bhat et al., 2018; Lloret et al., 2011): 1) Pre-Processing, 2) sentence extraction (extractive ATS phase): extract the key sentences from the input text (Wang et al., 2017), 3) summary generation (abstractive ATS phase): generate the final abstractive summary by applying the abstractive methods and techniques on the extracted sentences from the first phase, and 4) Post-Processing: to ensure that the generated sentences are valid, some general rules need to be defined like (Lloret, Romá-Ferri, & Palomar, 2013):

1. The minimal length for a sentence must be 3 words (i.e. subject + verb + object).
2. Every sentence must contain a verb.
3. The sentence should not end with an article (e.g. "a", and "the"), a preposition (e.g. "of"), a conjunction (e.g. "and"), nor an interrogative word (e.g. "who").

Advantages: Combining the advantages of both extractive and abstractive approaches. The two approaches are complementary and the overall performance of summarization is improved (Wang et al., 2017).

Disadvantages: Generating a less quality abstractive summary than the pure abstractive approach because the generated summary depends on the extracts instead of the original text.

The research community is focusing more on the extractive ATS approach using different methods and techniques, trying to

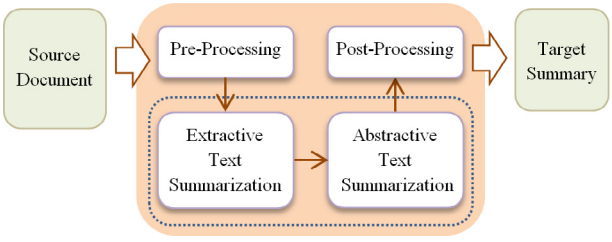

Fig. 6. The architecture of a hybrid text summarization system.

achieve more coherent and meaningful summaries (Gambhir & Gupta, 2017) because the abstractive approach is highly complex and needs extensive NLP.

3.3.2. Methods

Fig. 3 shows various hybrid text summarization methods. Table 3 illustrates the pros, cons, and examples of summarizers for each method. Up to our knowledge, there are mainly two methods that have been used in the hybrid text summarization: extractive to abstractive and extractive to shallow abstractive methods. Both methods are briefly discussed next.

Extractive to Abstractive Methods: These methods start by using one of the extractive ATS methods then they use one of the abstractive text summarization methods which is applied to the extracted sentences. In Wang et al. (2017), Wang et al. propose a hybrid system for long text summarization "EA-LTS". The system consists of two phases: 1) the extraction phase that uses a graph model to extract the key sentences, and 2) the abstraction phase that constructs an RNN based encoder-decoder and uses a pointer and attention mechanisms to generate summaries.

Extractive to Shallow Abstractive Methods: These methods start by using one of the extractive ATS methods then they use a shallow abstractive text summarization method that applies one or more of the following techniques to the extracted sentences: information compression techniques, information fusion techniques (Lloret et al., 2013), synonym replacement techniques (Patil, Dalmia, Ansari, Aul, & Bhatnagar, 2014), etc. In Bhat et al. (2018), Bhat et al. propose a single-document hybrid ATS system called "SumItUp". The hybrid system consists of two phases as follows:

1. Extractive Sentence Selection: uses some statistical features (sentence length, sentence position, TF-IDF, noun phrase and verb phrase, proper noun, aggregate cosine similarity, and cue-phrases) and a semantic feature (emotion described by text) to generate the summary. Cosine similarity is used to remove the redundant sentences in the extractive summary.
2. Abstractive Summary Generation: the extracted sentences are fed to a language generator (i.e. a combination of WordNet, Lesk algorithm and part-of-speech tagger) to convert the extractive summary to the abstractive summary. To retain the original sequence, sentences are reordered based on their initial index.

In conclusion, the hybrid summarization approach is a promising research direction. Mahajani et al. recommend researchers to propose hybrid ATS systems in order to benefit from the advantages of both extractive and abstractive approaches (Mahajani et al., 2019).

4. Techniques and building blocks to implement the ATS systems

This section provides an overview about the different components and techniques that are used to design and implement ATS systems. First, the text summarization operations are defined. These operations are defined based on the analysis of the human experts' operations. Second, the statistical and linguistic features are presented. These features are widely used to distinguish important words and sentences. Then, the text summarization building blocks are presented as follows: text representation models that are widely used to represent the input texts, linguistic analysis and processing techniques that are used in the different ATS phases, and the soft computing techniques that are useful in ATS implementations.

4.1. Text summarization operations

Text summarization operations can be classified into two categories (Belkebir & Guessoum, 2018): 1) single-sentence operations, and 2) multi-sentence operations. A single-sentence operation is applied to a single sentence and a multi-sentence operation is applied to two sentences at least. Text summarization operations can also be classified as either (Hasler, 2007): 1) atomic operation which cannot be further divided into other operations (e.g. insertion and deletion of words), and 2) complex operation which can be divided into other operations (e.g. replacement and reordering of words and merging of sentences). Fig. 7 shows the text summarization operations classified as either single-sentence or multi-sentence operations. Some operations can be used alone, in sequence, or in parallel to convert a source document into a summary document (Jing, 2002). Based on analyzing the operations of the human experts in (Jing, 2002), Jing defines the first 7 of the following operations:

1. Sentence Compression/Reduction: removal of unimportant parts (e.g. phrases) to shorten the original sentence.
2. Syntactic Transformation: transformation of a sentence by changing its syntactic structure (e.g. the position of the subject in a sentence may be moved from the end to the front). This operation may be used in both sentence compression and sentence combination operations.
3. Lexical Paraphrasing: replacement of phrases by their paraphrases.
4. Generalization: replacement of phrases or clauses by more general descriptions.
5. Specification: replacement of phrases or clauses by more specific descriptions.
6. Sentence Combination/Fusion: merge of two or more original sentences into a single summary sentence.
7. Sentence Reordering: change of the order of summary sentences (e.g. an ending sentence in an original text is placed at the beginning of the summary).
8. Sentence Selection: selection of one sentence from two or more similar sentences.
9. Sentence Clustering: grouping of the sentences into different clusters. This operation is very useful in multi-document summarization (e.g. identify the subject and cluster the sentences by subject (Zhong et al., 2017)).

Each ATS system uses one or more of the above operations. In the literature, there are many proposed techniques and algorithms

Table 3
Pros and cons of the hybrid text summarization methods.

Method	Pros	Cons	Examples
Extractive to Abstractive	Using both extractive and abstractive methods to improve the quality of the generated summary (Sahba, Ebadi, Jamshidi, & Rad, 2018).	The extractive method used in the first stage has an important effect on the final performance (Liu, Saleh, Pot, Goodrich, Sepassi, Kaiser, & Shazeer, 2018).	(Gehrmann, Deng, & Rush, 2018; Liu et al., 2018; Rudra et al., 2019; Sahba et al., 2018; Zeng, Luo, Fidler, & Urtasun, 2016)
Extractive to Shallow Abstractive	Enhance the disadvantages of the generated extractive summary.	The final summary is a shallow abstractive summary.	(Sahoo, Bhoi, & Balabantaray, 2018)

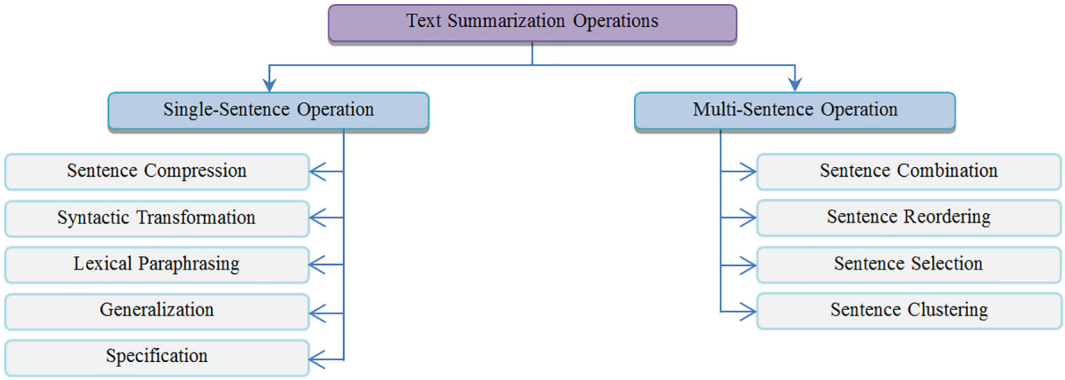

Fig. 7. Single-sentence and multi-sentence text summarization operations.

that perform these operations automatically like Linhares Pontes, Huet, Torres-Moreno, and Linhares (2020), Vanetik, Litvak, Churkin, and Last (2020) for sentence compression.

4.2. Statistical and linguistic features

The statistical and linguistic features are mainly used to identify the important sentences and phrases in the input document(s). Both the word level and sentence level features are used in the text summarization literature (Moratanch & Chitrakala, 2017). Many ATS systems in the literature are based on human-engineered features. Table 4 lists the most common features that are clues to include a sentence or phrase in the final summary. The simplest automatic text summarizer is based on the sentence selection such that sentences are ranked for potential inclusion in the summary using a weighted combination of statistical and linguistic features (Goldstein, Kantrowitz, Mittal, & Carbonell, 1999). This paradigm transforms the problem of summarization, which generally requires the ability to understand, interpret, abstract and generate a new summary document, into a simpler problem that is solved by the following steps (Goldstein et al., 1999): 1) assigning a score to each sentence based on the score of their features, and 2) concatenating the top-ranked sentences to form a summary (Wang et al., 2017). The sentence score is computed using Eq. (1).

$$\text{Score}(S_j) = \sum_{i=1}^{N} F_i(S_j) * W_i \quad (1)$$

Where S is the sentences list, j is the sentence's index within the sentences list, F is the features set, W is the list of weights values for these features, and N is the number of features. Defining proper weights for the features plays a major role in choosing the important sentences (Moratanch & Chitrakala, 2017) and hence affects the quality of the generated summary. The weights assigned to a feature may differ according to the type of summary and genre of corpus or document. These weights can be optimized for specific applications and genres. For each sentence "j" in the input document(s) sentences list "S", the sentence score can be normalized using Equation (2) by dividing this sentence score "Score(Sj)" by the maximum value of all sentences' scores "MaxScore(S)".

$$\textit{Normalized Score}(S_j) = \frac{\text{Score}(S_j)}{\text{MaxScore}(S)} \quad (2)$$

4.3. Text representation models/schemes

Many text representation models have been used to represent the input document(s) in the processing phase of the ATS systems. These text representation models are depicted in the left-hand side of Fig. 8. In the following, we briefly describe the most relevant text representation models.

4.3.1. Graph model

Graph theory has been successfully applied to represent the semantic contents of a document (Vodolazova et al., 2013b) or the document structure, so graph modeling is widely used in document summarization. In a graph, text elements (words or sentences) are represented by nodes and edges connect the related text elements together (e.g. semantic relation) (Gambhir & Gupta, 2017). There are two types of graphs to represent text: lexical graph and semantic graph.

Lexical Graph: uses the lexical features of the text to create a graph. TextRank (Mihalcea & Tarau) and LexRank (Erkan & Radev, 2004) are lexical graph-based systems. They create graphs by representing sentences as nodes and the edges connecting two sentences represent the degree of content similarity between them (Joshi et al., 2018). TextRank and LexRank are very similar. The key difference between them is that TextRank computes similarity as the number of similar words between two sentences, while Lexrank uses the cosine similarity between sentences and it is adjusted for multi-document summarization (Alami, El Adlouni, En-nahnahi, & Meknassi, 2018).

Semantic Graph: uses the semantic properties of the text. Semantic properties are the ontological relationship between two words (such as synonymy and hyponymy) and the relationship among a set of words representing the syntactic structure of sentences (such as the dependency tree and syntactic trees). The semantic relations between words are very important because a set of words and their way of arrangement provide a meaning. The same set of words arranged in different ways has a different meaning (Joshi et al., 2018).

4.3.2. Vector model

There are many models of vector representation like Bag-of-Words, Vector Space Model, and Word Embedding.

Bag-of-Words Model (BoW) (Harris, 1954): each sentence in the input text is represented as an N-dimensional vector, where (N-1) is the number of all possible words in the text language (Erkan & Radev, 2004). An entry "UNK" is added to the vector to represent any out-of-vocabulary words (Finegan-Dollak, 2018). For each word in the sentence, its entry value in the BoW vector is the number of occurrences of the word in the sentence multiplied by the IDF of the word (Erkan & Radev, 2004). The BoW representation has some weaknesses such as (Finegan-Dollak, 2018): 1) it does not include term dependency and has a high dimensionality (dos Santos et al., 2018), 2) it is very sparse because most of the values in the vector are zeros, 3) it ignores syntax (e.g. "The dog chased the cat" and "The cat chased the dog" have the same

Table 4
Statistical and linguistic features.

Feature	Explanation
Term Frequency (TF)	In single-document summarization, the TF of a word is the number of times it occurs in the document. In multi-document summarization, the TF of a word is computed by dividing the number of times it occurs in all the documents by the number of documents. TF helps to identify the most significant concepts or topics in the input document(s) by identifying the most frequent words (Lloret et al., 2011; Vodolazova, Lloret, Muñoz, & Palomar, 2013b).
Inverse Document Frequency (IDF)	IDF is used to measure how much information the word provides across all documents in the cluster. IDF values are close to zero for the common words like articles (e.g. "a" and "the") and IDF values are higher for rare words (e.g. medical terms and proper nouns) (Erkan & Radev, 2004). IDF is computed by dividing the total number of documents in a cluster by the number of documents that contain the word and then taking the logarithm of that quotient.
TF-IDF	The topics of the document cluster are identified by the words with higher TF-IDF values (Tandel et al., 2016). TF-IDF of a certain word is computed by multiplying the TF and IDF values of this word.
Code Quantity Principle (CQP)	The most important information within a text is expressed by a high number of units (i.e. words or noun phrases) (Lloret et al., 2011). CQP proves the existence of a proportional relationship between how important the information is, and the number of coding elements it has. A coding element can vary depending on the desired granularity (e.g. syllables or noun-phrases) (Lloret & Palomar, 2009).
Noun and Verb Phrases	If a sentence contains noun and verb phrases, it is considered an important sentence and is included in the generated summary as it contains valuable information (Kirmani et al., 2019).
Content Word (Keyword)	Key expressions that appear in an article (Takeuchi, 2002). Content words or Keywords are usually nouns and determined using the TF-IDF measure (Gupta & Lehal, 2010).
Title Word	The frequency of a keyword's appearance in a title or headline of an article (Takeuchi, 2002). The resemblance of a sentence to the title (Gambhir & Gupta, 2017). Sentences containing words that appear in the title indicate the theme of the document (Gupta & Lehal, 2010).
Proper Noun	Presence of a proper noun (name entity) in the sentence (Gambhir & Gupta, 2017). Sentences containing proper nouns (e.g. name of a person, place, and concept) have greater chances to be included in the summary.
Cue-Phrase	Sentences containing cue phrases are most likely to be in the summary: "argue", "in conclusion", "this letter", "this paper", "summary", "purpose", "develop", "propose", "attempt", etc. (Gupta & Lehal, 2010; Vodolazova et al., 2013b)
The occurrence of Non-Essential Information	Some words are indicators or markers of non-essential information like: "because", "furthermore", and "additionally". These words usually occur at the beginning of a sentence. This is a binary feature: the value is "true" if the sentence contains at least one of these words, and "false" otherwise (Gupta & Lehal, 2010).
Biased Word	The sentence is important if it includes one word or more from a biased word list (i.e. a predefined list that may contain domain-specific words) (Gupta & Lehal, 2010).
Font based	Sentences containing words that appear in upper case (e.g. acronyms or proper names), italics, bold, or underlined fonts are usually more important and are included in the summary (Gupta & Lehal, 2010).
Positive Keyword	The frequency count of positive words in the sentence (Gambhir & Gupta, 2017).
Negative Keyword	The frequency count of negative words in the sentence (Gambhir & Gupta, 2017).
Numerical Data	Presence of numerical data in the sentence (Gambhir & Gupta, 2017).
Sentence Location (Position of Sentence)	The position of a sentence in an article or in a paragraph (Takeuchi, 2002). Sentences located in the beginning or end of the text are usually considered to be more important and are selected for the final summary (Gupta & Lehal, 2010; Vodolazova et al., 2013b).
Sentence Length	The relative length of the sentence. Sentence with the length below some predefined threshold can be automatically ignored (Vodolazova et al., 2013b). Very short and very long sentences are usually not selected in the summary (Gupta & Lehal, 2010).
Sentiment Features (Emotions described by Text)	Emotion described by text is a semantic feature. Sentences that include implicit emotional content are important to the writer and should be added to the summary (Bhat et al., 2018). There is several emotion classes that are used to define emotions in the input text: positive, negative, joy, fear, hate, surprise, disgust, etc. (Kirmani et al., 2019).
Sentence Reference Index (SRI)	SRI gives more weight to a sentence that precedes a sentence containing a pronominal reference. Using a list of pronouns, if a sentence contains a pronoun then the weight of the preceding sentence is increased (Gupta & Siddiqui, 2012).
Sentence-to-Sentence Cohesion	The similarity with other sentences (Gambhir & Gupta, 2017). For each sentence S, compute the similarity between S and each of the other document sentences then sum all these similarity values to get this feature value for S (Gupta & Lehal, 2010).
Sentence-to-Centroid Cohesion	For each sentence S, compute the vector representing the centroid of the document (i.e. the arithmetic average over the corresponding coordinate values of all the sentences of the document) then compute the similarity between the centroid and each sentence to get this feature value for each sentence (Gupta & Lehal, 2010).
Concept Similarity of a Sentence	The concept similarity of a sentence is the number of synonym sets (synsets) of query words matching with words in the sentence. The set of synsets obtained from WordNet (Wordnet, 2020) is used to assign concept similarity weight to a sentence (Gupta & Siddiqui, 2012).

vectors), 4) it ignores the meaning of words (e.g. the dogs and cats are animals and the chasing is a type of movement), and 5) inaccurate semantic representation of text because word order and arrangement are ignored (Kim, Park, Lu, & Zhai, 2012).

Vector Space Model (VSM) (Salton, Wong, & Yang, 1975) **or Term Vector Model**: it is a classical form of document representation that represents text documents as vectors (Ibrahim, Elghazaly, & Gheith, 2013). VSM is commonly used in Information Retrieval (IR) by representing documents and queries as vectors of weights. A document d consists of a set of terms ($t_1, t_2 \ldots t_n$) where each term is a word that exists in the document, and n is the total number of different words. Each term t has a corresponding weight w (Mingzhen & Yu, 2009) such that each weight measures the significance of the term in the document or query respectively. The weights are computed based on the frequency of the terms in the query or the document (e.g. TF-IDF weights) (Melucci, 2009). The great advantage of VSM is the simplicity to search for documents or compare documents by using one of the similarity or distance measures like the Spearman distance, cosine measure, Euclidean distance, Vector inner product, Hamming distance, Correlation distance, or Jaccard distance (Mingzhen & Yu, 2009). At retrieval time, the documents are ranked based on the cosine similarity values between the document vectors and the query vector and returned to the user from the highest to the lowest values (Melucci, 2009). The same concept is used for query-based text summarization and clustering the similar sentences where text parts (e.g. title, query, and sentences) are represented as vectors. The bag-of-words model is the most popular VSM approach due to its simplicity and general applicability (dos Santos et al., 2018).

Word Vector or Word Embedding: neural networks are used for learning vector representations of words. For example, different methods are used to get the word embedding (Mogren, Kageback,

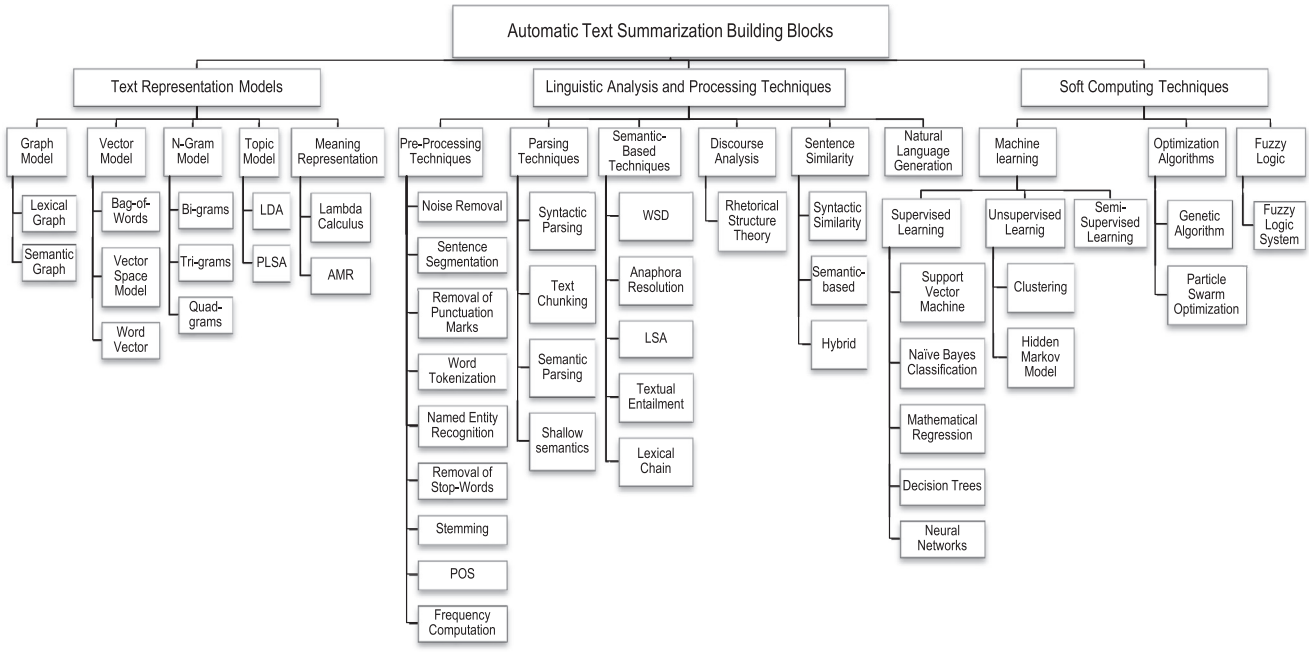

Fig. 8. Building blocks of the ATS systems.

& Dubhashi, 2015) like the Word2Vector model proposed by Mikolov et al. in Mikolov, Chen, Corrado, and Dean (2013). Each word embedding (i.e. distributed representation of a word) is a real-valued vector in the Euclidean space that corresponds to the "meaning" of the word (Kobayashi et al., 2015). The word vectors are averaged or concatenated to predict the next word in a context (Wang et al., 2017). Recently, new related types of text representations have emerged such as sentence, paragraph, or document embeddings.

4.3.3. N-gram model

N-gram is perfect for multi-language operations because it requires no linguistic preparations (e.g. stop word removal or stemming). N-gram is a set of words or characters that contains N elements. Word N-grams are a sequence of one word (uni-grams), two words (bi-grams), three-word (tri-grams) or any other N-grams (Abdolahi & Zahedh, 2017). Each word can be represented as a set of character N-grams. For example, the word "TEST" could be represented with the following N-grams (Cavnar, 1994):

1. bi-grams: _T, TE, ES, ST, T_
2. tri-grams: _TE, TES, EST, ST_
3. quad-grams: _TES, TEST, EST_

Where the underscore character represents a leading or tailing space. Similar words will share a large number of character N-grams. For example, the words "RETRIEVE", "RETRIEVING", and "RETRIEVAL" share the bi-grams: _R, RE, ET, TR, RI, IE, and EV.

4.3.4. Topic model

A document is represented as a combination of topics and each topic is a probability distribution over words (Singh, Devi, & Mahanta, 2017). A topic model is a generative probabilistic model that can be used to identify topics of the input text represented as word distributions. A word distribution represents a topic by appointing high probabilities to words that portray a topic (e.g. in reviews of iPhone, a topic about battery life may have high probabilities for words like "hour", "battery", and "life") (Kim et al., 2012). The two basic representative topic modeling approaches are Latent Dirichlet Analysis (LDA) (Blei, Ng, & Jordan, 2003; Kim et al., 2012) and Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999).

4.3.5. Meaning representations (MR) model

Commonly used general-purpose meaning representations include Lambda Calculus and Abstract Meaning Representation.

Lambda Calculus: represents meanings as functions applied to variables and constants. For example: the sentence "what states border Texas" may be represented as λx: state(x) ^ borders(x; Texas), x is a variable, state() is a function that applies to a single entity, and borders() is a function that defines a relationship between two entities. Lambda calculus can use higher-order functions, which are functions that take other functions as inputs. It can incorporate quantifiers and represent all of the first-order logic. Lambda calculus is not easy to write, so using it to generate anything more than toy datasets is difficult (Finegan-Dollak, 2018).

Abstract Meaning Representation (AMR): represents the sentences as rooted directed graphs with labels for the edges and leaf nodes. It integrates PropBank frames, modality, co-reference, reification, negation, and other concepts from a variety of views on what is important in semantics. Most of the research on AMR has focused on the parsing of the English sentences to AMR. Recently, many research work explored the usefulness of AMR in many applications such as ATS, headline generation, machine comprehension, and question answering (Finegan-Dollak, 2018).

4.4. Linguistic analysis and processing techniques

The linguistic properties of the original document affect the quality of the generated summary (Vodolazova et al., 2013b). There are many linguistic analysis and processing techniques that are widely used in ATS (Vodolazova et al., 2013b). These techniques are also illustrated in near the center of Fig. 8 under the "Linguistic Analysis and Processing Techniques" subtree. Stanford CoreNLP

(Manning, Surdeanu, Bauer, Finkel, Bethard, & McClosky, 2014) is a commonly used tool that provides many of the common core NLP steps like words tokenization and co-reference resolution. In the following, we briefly describe the most relevant linguistic analysis and processing techniques.

4.4.1. Pre-processing techniques

There are many pre-processing techniques like noise removal, sentence segmentation, word tokenization, etc. Most of these techniques are usually used in the pre-processing phase of an ATS system.

Noise Removal: removes unnecessary text from the input document like the header, footer, etc. (Gupta & Siddiqui, 2012)

Sentence Segmentation: divides the text into sentences (Gambhir & Gupta, 2017). Splitting sentences by using end markers like ".", "?", or "!" is not suitable in many cases. If we rely on these markers, words like "e.g.", "i.e.", "4.5", "Mr.", "Dr.", or "etc." lead to the false identification of sentence boundaries. To solve this problem, simple heuristics and regular expressions are used (Gupta & Siddiqui, 2012).

Removal of Punctuation Marks: punctuation marks are considered as noisy terms in the text. So, removing them is very helpful before executing most NLP tasks (Gambhir & Gupta, 2017).

Word Tokenization: breaks the text into separate words. Words are separated by white space, comma, dash, dot, etc. (Gupta & Siddiqui, 2012)

Named Entity Recognition (NER): identifies words of the input text as names of things (i.e. person name, location name, company name, etc.).

Removal of Stop-Words: stop-words are words that occur frequently in the text like articles, pronouns, prepositions, auxiliary verbs, and determiners. They are removed because they do not add any useful meaning to the analysis (Jaradat & Al-Taani, 2016) and have no effect in selecting the important sentences (Gambhir & Gupta, 2017).

Stemming: reduces the words with the same root or stem to a common form by removing the variable suffixes (Gambhir & Gupta, 2017; Manning, Raghavan, & Schütze, 2008) like "es" and "ed". The purpose of stemming is to obtain the stem or radix of each word to put emphasis on its semantics (Gupta & Lehal, 2010).

Part-Of-Speech (POS) Tagging: assigns POS tags (e.g. verb, noun, etc.) to words in a sentence.

Frequency Computation: the frequency of words is computed and normalized by dividing it by the maximum frequency of any word in the document (Gupta & Siddiqui, 2012). Most frequent words are very important to help in the selection of the most important sentences in the original document(s). By analyzing the human-made summaries, they are very likely to include the high-frequency words from the original documents (Lloret & Palomar, 2009).

4.4.2. Parsing techniques

There are several uses of parsing techniques in the ATS processing phase like constructing the text graph models and in the post-processing phase like sentence compression, sentences merging, etc. There are many parsing techniques like syntactic parsing, text chunking, semantic parsing, and shallow semantics.

Syntactic Parsing: refers to the task of recognizing a sentence and assigning a syntactic structure to it. The standard way to represent the syntactic structure of a grammatical sentence is as a syntax tree (or a parse tree) which is a representation of all the steps in the derivation of the sentence from the root node (i.e. each internal node in the tree represents an application of a grammar rule) (Indurkhya and Damerau, 2010). Parse trees are useful in grammar checking: a sentence that cannot be parsed may have grammatical errors (or at least is hard to read) (Jurafsky & Martin, 2017). Constructing the syntactic structure of a sentence is important to understand it. Without this, it is very challenging for language users to specify that sentences with the same words and different word orders have different interpretations (e.g. "The woman sees the man" and "The man sees the woman") or explain why a sentence like "The hunter killed the poacher with the rifle" has two possible interpretations (Levelt & Caramazza, 2007).

Text Chunking: refers to a complete partitioning of a sentence into chunks of different types like verb groups, noun groups, etc. Full parsing is not very robust and expensive hence text chunking plays an important role in NLP. Partial parsing is more robust, much faster, and sufficient for many applications like question answering, information extraction, etc. Besides, it can be used as the first step for the full parsing process. Text chunking is considered as a shallow parsing technique (Maiti, Garain, Dhar, & De, 2015).

Semantic Parsing: refers to the task of converting natural language text to a complete and formal meaning representation (Clarke, Goldwasser, Chang, & Roth, 2010). Some researchers have focused on general-purpose meaning representations such as lambda calculus, AMR, and Prolog, while others have focused on more task-specific meaning representations (Finegan-Dollak, 2018).

Shallow Semantics: represents just a small portion of the semantic information about the sentence. Semantic Role Labeling is a main example. SRL labels the constituents of a sentence with their semantic roles (e.g. to identify the agent and patient of a verb). In the labeled sentence S1 "[John $_{\text{AGENT}}$] ate [the fish $_{\text{PATIENT}}$]-", John is the agent who did the eating. The sentence S2 "[The fish $_{\text{AGENT}}$] ate [John $_{\text{PATIENT}}$]" has a different meaning as the fish did the eating. The sentence S3 "[The fish $_{\text{PATIENT}}$] was eaten by [John $_{\text{AGENT}}$]" is syntactically different from S1 while being semantically equivalent to it. As such S3 is a paraphrasing of S1 (Finegan-Dollak, 2018).

4.4.3. Semantic-based techniques

There are many semantic-based techniques like word sense disambiguation, anaphora resolution, etc. The use of these techniques covers all the ATS system phases as follows: 1) in the pre-processing, textual entailment and word sense disambiguation techniques are used, 2) in the processing phase, the latent semantic analysis and lexical chain techniques are used, and 3) in the post-processing phase, word sense disambiguation, anaphora resolution, and textual entailment techniques are used.

Word Sense Disambiguation (WSD): identifies the proper meaning of the given word (i.e. computationally find the correct sense of the ambiguous words using the context in which they occur). For example: the word "bank" has many meanings in English. Such words with multiple meanings are called polysemous words. WSD is the process of finding out the exact meaning of the polysemous word (Sheth, Popat, & Vyas, 2018).

Anaphora Resolution: analyzes the pairs of nouns, pronouns, and proper nouns in a document. A powerful anaphora resolution tool relates pronouns to their nominal antecedents. It is used in all the summarization methods that depend on term overlap, from the simple term frequency to latent semantic analysis (Vodolazova et al., 2013b). A common problem in the extractive ATS is the anaphora "dangling" in the sentences that contain pronouns while missing their referents when extracted out of context. Moreover, stitching together decontextualized extracts may mislead the interpretation of anaphors hence cause an inaccurate representation of the source information (Gupta & Lehal, 2010).

Latent Semantic Analysis (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990): excerpts hidden semantic structures of words and sentences. LSA is an unsupervised learning approach so it does not require any training data. In LSA, a term-by-sentence matrix representation is created from the input

document(s). LSA gets information such as words that frequently occur together and words that are commonly used in different sentences. A high number of common words among the sentences means that the sentences are semantically related. SVD is a method that is applied to the term-by-sentence matrix. SVD is used to find out the interrelations between words and sentences which has the competence of noise reduction that helps to improve accuracy. SVD is applied to document word matrices to group documents that are semantically related despite lacking common words. The set of words that result in the connected text is also connected in the same dimensional space (Moratanch & Chitrakala, 2017).

Textual Entailment (TE): determines if the meaning of one text snippet (the hypothesis) can be inferred by another one (the text) (Lloret & Palomar, 2009) so it is used to capture the semantic inference between text fragments (Vodolazova et al., 2013b). If two sentences contain a true entailment relationship, they are considered equivalent and the one which is entailed can be discarded (Lloret et al., 2011). There are different uses of the TE in ATS systems: 1) in the process of the final summary evaluation (i.e. in a set of generated summaries, TE is used to decide which summary of them can be best deduced from the original document), 2) in the segmentation of the input by using different algorithms that employ TE, and 3) in the process of summary generation to avoid redundant information appearing in the final summary (Lloret & Palomar, 2009). TE is often applied to eliminate the semantic redundancy of the generated summary (Vodolazova et al., 2013b).

Lexical Chain: represents the semantic content that may cover a small or big part of the text. The coverage and size of a lexical chain indicate the accuracy of the lexical chain to represent the semantic content of the text. A lexical chain contains a set of words (word senses) that are semantically related in the text. Lexical chains are constructed by using relationships between word senses. So it is required to know the word senses and semantic relations between words. WordNet is a database that provides this type of information: synonym sets, hyponym/hypernym, and meronym trees. For each lexical chain, the number of semantic relations and the number of words among the words can be different. Building a lexical chain is an exhaustive method because one word can have many senses and it is required to select its correct sense of the word. Lexical chains are used in text summarization and keyword extraction (i.e. keywords are short forms and condensed versions of the documents and their summaries) (Ercan & Cicekli, 2007).

4.4.4. Discourse analysis

Discourse relations in the text represent connections between sentences and parts in a text (Gambhir & Gupta, 2017). It is essential to determine the overall discourse structure of the text for producing a coherent and fluent summary (i.e. contains the flow of the author's argument) then being able to remove unrelated sentences to the context of the text (Gupta & Lehal, 2010). Mann and Thompson proposed the Rhetorical Structure Theory (RST) (Mann William & Thompson Sandra, 1988) to act as a discourse structure. RST is one of the well-known models for text structure representation and is mainly used to represent the coherence of texts. Using RST, text can be divided into sub-parts forming a hierarchical structure. Every sub-part has a relationship to another sub-part with one of the rhetorical relation types (e.g. Motivation, Contrast, and Elaboration). These relations form the overall coherence structure of the text. A small number of defined rhetorical relations can be used to explain the relationships among a wide range of texts (Takeuchi, 2002). RST has two main concepts (Gambhir & Gupta, 2017): 1) coherent texts contain few units connected together by rhetorical relations, and 2) there must be some relations between various parts of the coherent texts.

4.4.5. Sentence similarity

The similarity techniques are widely used to measure the similarity between sentences or texts in general and can be classified into three categories (Wali, Gargouri, & Ben Hamadou, 2017): syntactic similarity, semantic similarity, or hybrid methods. In the following, we briefly describe each of these.

Syntactic Similarity Methods: string matching, word order, and word co-occurring are counted to compute the syntactic similarity. Calculating the co-occurring words in a string sequence may not always be useful because sentences may be very similar while having rare co-occurring words. For literal similarity, Levenshtein distance (edit distance) is a string metric that can be used for measuring the difference between two sequences. Levenshtein distance between two words is the minimum number of single-character edits (e.g. insertions, deletions, or substitutions) required to change one word to the other (Wali et al., 2017; Wang et al., 2017).

Semantic Similarity Methods: use the semantic nets like WordNet, the vector space model, and the statistical corpus to compute the semantic similarity between words using different known measures. The semantic-based methods are limited to compute the sentence similarity based only on the semantic similarity between words. The syntactic information and other semantic knowledge (e.g. semantic class and thematic roles) are not used. For example: after training the sentence vectors, they are used as features for the sentences. The semantic similarity between two sentences is simply obtained by calculating the cosine similarity of their vectors (Wali et al., 2017; S. Wang et al., 2017).

Hybrid Methods: use both semantic and syntactic knowledge. The main disadvantage of these hybrid methods is that the semantic measurement is isolated from the syntactic measurement (Wali et al., 2017).

4.4.6. Natural language generation

Natural Language Generation (NLG) is the task of generating texts from the input information like texts, knowledge, or images. It is a challenging task because it requires understanding the input information then organizing the text that will be generated. Simply, NLG requires to answer the following questions: "what to say?" and "how to say it?" (Li, Sun, & Li, 2019). Generating good texts is highly dependent on the good representation and understanding of the input information. NLG has been used in many applications like text summarization, descriptions of museum artifacts, auto-completion, dialog systems, auto-paraphrasing, question answering, machine translation (Kurup & Narvekar, 2020). NLG systems are widely used in the abstractive text summarization methods to generate the final abstractive summary with sentences different than the original text sentences.

4.5. Soft computing techniques

Soft computing solves complex problems by manipulating the uncertainty and imprecision in the decision making practices. Soft computing is guided by a main principle of "exploit the tolerance for imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost" (Rao & Svp Raju, 2011). Many soft computing techniques have been used as building blocks in the ATS systems such as machine learning algorithms, fuzzy logic system, genetic algorithm, etc. All these techniques are not competitive but complementary to each other, so each of these techniques can be used individually or with each other to get better summarization results (Ibrahim, 2016). For example, "neurofuzzy" system is an effective combination that merges between the neural networks and fuzzy logic techniques (Rao & Svp Raju, 2011). This survey cannot cover all soft computing

techniques because there exists a huge number of algorithms. Only the most commonly used ones are illustrated in the right-hand side of Fig. 8. In the following, we provide a brief description of each.

4.5.1. Machine learning algorithms

Machine learning algorithms are widely used in ATS systems like Alguliyev, Aliguliyev, Isazade, Abdi, and Idris (2019), Shetty and Kallimani (2017), Yousefi-Azar and Hamey (2017). Machine learning algorithms are categorized as: supervised, unsupervised, or semi-supervised.

Supervised Learning Algorithms: require a large amount of labeled or annotated data as input to a training stage (Moratanch & Chitrakala, 2017). Commonly-used supervised learning algorithms include: Support Vector Machine (SVM), Naïve Bayes Classification, Mathematical Regression, Decision Trees, and Artificial Neural Networks (ANN).

Unsupervised Learning Algorithms: do not require any training data. They try to discover the hidden structure in unlabeled data. These techniques are therefore suitable for any newly observed data without any required modifications. Commonly-used techniques of that type include clustering, and Hidden Markov Model (HMM). When used for ATS, these systems access only the target documents and apply heuristic rules to extract highly relevant sentences to generate a summary.

Semi-Supervised Learning Algorithms: require both labeled and unlabeled data to generate an appropriate function or classifier.

4.5.2. Optimization algorithms

Optimization algorithms have been widely used for ATS systems like Sanchez-Gomez, Vega-Rodríguez, and Pérez (2018) and Alguliyev et al. (2019). The most common algorithms are the genetic algorithm and particle swarm optimization.

Genetic Algorithm (GA): A GA is a search based optimization algorithm inspired by the analogy of evolution and population genetics. The GA is effective in searching for very large and varied spaces in a wide range of applications (Jaradat & Al-Taani, 2016). It optimizes initially-random solutions by applying natural evolution operations like: selection, mutation, and crossover on them (Gambhir & Gupta, 2017). Al-Radaideh and Bataineh (2018), Chatterjee, Mittal, and Goyal (2012), García-Hernández and Ledeneva (2013), Neri Mendoza, Ledeneva, and García-Hernández (2019) show examples of ATS research making use of GAs. The genetic algorithm steps include (Ibrahim, 2016):

1. Initialization: creating an initial population randomly.
2. Evaluation: evaluating each member of the population and assessing its fitness based on how close it matches the preferred requirements.
3. Selection: selecting some individuals while favoring those with higher fitness.
4. Crossover: creating new individuals by combining the features of the selected individuals. At the end of this step, it is expected that the created individuals are closer to the preferred requirements.
5. Repeat steps 2 to 5 until the termination condition is reached.

Particle Swarm Optimization (PSO): PSO is one of the most powerful bio-inspired algorithms used to obtain an optimal solution (Dalal & Malik, 2018). PSO algorithm is inspired by the social movement of birds (Nazari & Mahdavi, 2019). Al-Abdallah and Al-Taani (2017), Mandal, Singh, and Pal (2019), Priya and Umamaheswari (2019) are examples of ATS research making relying on the PSO algorithm. The PSO algorithm steps include (Venter & Sobieszczanski-Sobieski, 2003):

1. Starting with an initial population of particles (individuals) which are randomly discovered through the design space. Each individual has a random position and a velocity.
2. Calculating a velocity vector for each individual.
3. Updating the position of each individual using its previous position and the newly updated velocity vector.
4. Until convergence, repeating the above steps from the second step.

4.5.3. Fuzzy logic

Fuzzy logic has been widely used in ATS systems like Abbasi-ghalehtaki, Khotanlou, and Esmaeilpour (2016), Jafari et al. (2016), and Patel, Shah, and Chhinkaniwala (2019). The inputs of text features that are given to the fuzzy logic system include: sentence length, sentence similarity, etc. (Moratanch & Chitrakala, 2017). Fuzzy logic systems mainly contain four components (Ibrahim, 2016) as follows:

- Fuzzifier (fuzzification interface): it transforms the crisp input value to a fuzzy linguistic value because the input values are always crisp numerical values.
- Inference Engine: it uses the fuzzy inputs and the fuzzy rules to generate the fuzzy outputs.
- Fuzzy Knowledge Base: it contains the fuzzy rules in the form of "IF-THEN" rules including the linguistic variables.
- Defuzzifier (defuzzification interface): it is the last step of a fuzzy logic system. It converts the fuzzy outputs to crisp output actions.

5. Text summarization datasets and evaluation metrics

This section provides an overview about the basic resources that are used to evaluate and compare ATS systems. These resources include the well-known and standard datasets besides the manual criteria and automatic summary evaluation tools.

5.1. Standard datasets

In Dernoncourt et al. (2018), Dernoncourt et al. provide an overview of many corpora that have been used in the summarization tasks. This survey presents the most common benchmarking datasets which have been used for the ATS systems evaluation as shown in Table 5 including:

1. Document Understanding Conference (DUC) Datasets: these datasets are provided by the National Institute of Standards and Technology (NIST) and they are the most common and frequently used datasets in the text summarization research. The DUC corpora were released as part of the summarization shared task hosted at the DUC conference. The last DUC challenge was held in 2007. The DUC website contains datasets for DUC 2001 through DUC 2007. Each dataset contains both documents and their summaries in three forms: 1) manually created summaries, 2) automatically created baseline summaries, and 3) automatically created summaries that were generated by challenge participants systems. To get access to these datasets, it is required to complete some application forms that exist on the DUC website.[1] These datasets are usually used to evaluate the ATS systems but they do not provide enough data to train the neural networks models (Lin & Ng, 2019).
2. Text Analysis Conference (TAC) Datasets: in 2008, DUC became a summarization track in the TAC. It is required to

[1] https://www-nlpir.nist.gov/projects/duc/data.html.

Table 5
Standard datasets for text summarization.

	Dataset Name	Number of Documents	Language	Domain	Single-Document	Multi-Document	URL
1	DUC 2001	60 × 10	English	News	✓	✓	https://www-nlpir.nist.gov/projects/duc/data.html
2	DUC 2002	60 × 10	English	News	✓	✓	
3	DUC 2003	60 × 10, 30 × 25	English	News	✓	✓	
4	DUC 2004	100 × 10	Arabic, English	News	✓	✓	
5	DUC 2005	50 × 32	English	News	✗	✓	
6	DUC 2006	50 × 25	English	News	✗	✓	
7	DUC 2007	25 × 10	English	News	✗	✓	
8	TAC 2008	48 × 20	English	News	✗	✓	https://tac.nist.gov/data/index.html
9	TAC 2009	44 × 20	English	News	✗	✓	
10	TAC 2010	46 × 20	English	News	✗	✓	
11	TAC 2011	44 × 20	English	News	✗	✓	
12	EASC	153	Arabic	News, Wikipedia	✓	✗	https://www.lancaster.ac.uk/staff/elhaj/corpora.htm
13	SummBank	40 × 10	Chinese, English	News	✓	✓	https://catalog.ldc.upenn.edu/LDC2003T16
14	Opinosis	51 × 100	English	Reviews	✗	✓	http://kavita-ganesan.com/opinosis-opinion-dataset/
15	LCSTS	2,400,591	Chinese	Blogs	✓	✗	http://icrc.hitsz.edu.cn/Article/show/139.html
16	CAST	147	English	News	✓	✗	http://clg.wlv.ac.uk/projects/CAST/corpus/index.php
17	CNN-corpus	3,000	English	News	✓	✗	By email request through (Lins, Oliveira, et al., 2019)
18	Gigaword 5	9,876,086	English	News	✓	✗	https://catalog.ldc.upenn.edu/LDC2011T07
19	CNN/Daily Mail	312,084	English	News	✓	✗	https://github.com/deepmind/rc-data/

complete some application forms that exist on the TAC website[2] in order to get access to the TAC datasets.

3. Essex Arabic Summaries Corpus (EASC) Dataset (El-Haj, Kruschwitz, & Fox, 2015): it contains Arabic articles and the human-generated extractive summaries of these articles. EASC uses copyrighted material. It is the responsibility of the dataset users to comply with all associated copyright rules.
4. SummBank Dataset (Radev, Teufel, Saggion, Lam, Blitzer, Qi, & Drabek, 2003): it contains 40 clusters of news, human-written non-extractive summaries, 360 multi-document, and approximately two million multi-document and single document extracts produced by manual and automatic methods.
5. Opinosis Dataset (Ganesan et al., 2010): it contains 51 files. Each file is about a feature of a product (e.g. the battery life of iPod) that includes a set of reviews written by customers who bought that product. So, the dataset represents 51 topics such that each topic includes approximately 100 sentences. The dataset contains 5 manually written "gold" summaries for each topic. For most topics, the 5 summaries are different.
6. Large-scale Chinese Short Text Summarization (LCSTS) Dataset (Hu, Chen, & Zhu, 2015): It contains more than 2 million short texts with short summaries. This dataset is created from the SinaWeibo website (i.e. a Chinese microblogging website).
7. Computer-Aided Summarization Tool (CAST) Corpus (Hasler, Orasan, & Mitkov, 2003): it contains a set of newswire texts which are taken from the Reuters Corpus[3] and a few science texts from the British National Corpus.[4] The news texts part of the corpus is available after signing the agreement with Reuters[5] while the other part cannot be distributed. The corpus contains three types of information annotation: the sentence importance (essential or important), links between sentences, and text fragments which can be removed from the marked sentences. A sentence is considered unimportant if it is not annotated. This dataset could be very useful for developing sentence reduction and sentence selection algorithms.
8. CNN-corpus Dataset (Lins, Oliveira, et al., 2019): it can be used for single-document extractive summarization. It contains the original texts, highlights, and gold-standard summaries. This corpus was recently used in the extractive text summarization competition "DocEng'19" (Lins, Mello, & Simske, 2019). For research purposes, this corpus with all its annotated versions is freely available by requesting it from its authors.
9. Gigaword 5 Dataset: It is a popular dataset for abstractive summarization research. It contains ten million English news documents approximately so it is suitable for training and testing the neural networks models. Gigaword is criticized because it contains only headlines as summaries (Lin & Ng, 2019; Nallapati, Zhou, Santos, Gulcehre, & Xiang, 2016).
10. CNN/Daily Mail Corpus (Hermann et al., 2015): this corpus is used for the passage-based question answering task then it has been widely used for evaluating the ATS systems. (Nallapati et al., 2016) provide a modified version of this corpus such that it contains multi-sentence summaries for abstractive summarization evaluation.

In Table 5, the following features are defined for each dataset: 1) the dataset name, 2) the number of documents, 3) the language of data, 4) the domain of data (e.g. news or blogs), 5) whether the dataset supports single-document and/or multi-document summarization, and 6) the dataset URL. In Table 5, the first 5 features have been filled from Dernoncourt et al. (2018) except for the datasets "EASC", "SummBank", "CAST" and "CNN-corpus" as their features are extracted from their corresponding papers and websites. The number of documents for a multi-document summarization dataset is written like "30 × 10" which means that the dataset includes 30 clusters of documents and each cluster contains 10 documents approximately.

[2] https://tac.nist.gov/data/forms/index.html.
[3] http://about.reuters.com/researchandstandards/corpus/.
[4] http://www.natcorp.ox.ac.uk/.
[5] http://about.reuters.com/researchandstandards/corpus/how_to_apply.asp.

In conclusion, there is a need for more datasets that 1) support non-English languages, and 2) cover the various data domains for all languages because most of the available datasets focus on the news domain as shown in Table 5. In addition, Dernoncourt et al. conclude that 1) there is a need for more large-scale corpora especially for evaluating machine-learning-based and deep-learning-based summarization systems, and 2) a data standard is required for all summarization corpora because each corpus is organized differently (Dernoncourt et al., 2018). If the researchers evaluate their proposed ATS systems on many corpora, they will consume a lot of time. As a result, the research papers in the ATS field usually use one or very few corpora.

5.2. Summary evaluation

In the past two decades, there were many efforts to solve summary evaluation issues. NIST leads the effort by organizing the DUC and TAC challenges (Lloret, Plaza, & Aker, 2017). In Huang, He, Wei, and Li (2010), Huang et al. formulate four main objectives that should be considered to generate a condensed and readable summary:

1. Information Coverage: the summary should contain the important information of the input document(s).
2. Information Significance: the summary should cover the various topics of the input document(s). The most important topics can either be the main topics in the input document(s) as in the generic summarization or the user-preferred topics as in the query-based summarization.
3. Information Redundancy: minimize the redundant or duplicate information in the generated summary.
4. Text Coherence: the summary is not just a set of important but disconnected phrases or words. The summary should be readable and understandable text.

There are two evaluation measures to evaluate the generated summaries (Gupta & Lehal, 2010): 1) intrinsic methods: measure summary quality using human evaluation. The intrinsic evaluation assesses the coherence and the content coverage or informativeness of a summary (Lloret et al., 2017), and 2) extrinsic methods: measure summary quality through a task-based performance measure such as the information retrieval-oriented task. The extrinsic evaluation assesses the utility of summaries in a given application context (e.g. relevance assessment, reading comprehension, etc.) (Lloret et al., 2017). There are two ways of text summarization evaluation: manual and automatic. Summary evaluation is a very challenging issue in the text summarization research field. The automatically generated summaries have to be evaluated in order to assess the quality of the ATS systems that generated them (Lloret et al., 2017). The ATS system performance is usually compared to different baseline systems such as using leading sentences from the input document or using common text summarizers such as LexRank (Erkan & Radev, 2004), TextRank (Mihalcea & Tarau, 2004), MEAD (Radev, Blair-Goldensohn, & Zhang, 2001), etc.

5.2.1. Manual evaluation of summaries

The human judges may be asked to evaluate the computer-generated summaries using some or all of the following quality metrics (Lloret et al., 2017; Mani, 2001):

- Readability: assess the linguistic quality of the summary by checking that it does not contain gaps in its rhetorical structure or dangling anaphora.
- Structure and Coherence: the summary has to be well-organized and well-structured. It consists of a set of coherent and related sentences.
- Grammaticality: the summary should not include incorrect sentences that violate the grammar rules or capitalization errors.
- Referential Clarity: if the summary includes a pronoun, so the reader should identify the noun phrase it refers to easily.
- Content coverage: the summary should include the various topics that have been discussed in the input document(s).
- Conciseness and Focus: each sentence in the summary should enclose information that is related to the other sentences.
- Non-redundancy: summary should not include unnecessary repetition which may take different forms like: whole sentences or parts of sentences that are repeated, or the repeated use of a noun phrase or noun like "Jack Tomson" when a pronoun "he" is sufficient.

A qualitative evaluation like in Lloret et al. (2013) may be used to measure the user satisfaction by focusing on a five-point scale (1 = strongly disagree, 2 = disagree, 3 = neither agree nor disagree, 4 = agree, and 5 = strongly agree) that are used to answer the following questions:

- Q1: The summary reflects the most important issues of the document.
- Q2: The summary allows the reader to know what the article is about.
- Q3: After reading the original abstract provided with the article, the alternative summary is also valid.

As there is no best reference summary among the human-created model summaries, the Pyramid method (Nenkova & Passonneau, 2004; Nenkova, Passonneau, & McKeown, 2007) was created to overcome this problem. This method is semi-automatic and has been used in DUC 2006 to evaluate summary content. The main idea is to create a gold standard summary by comparing the human-created reference summaries based on Summary Content Units (SCUs). First, the similar sentences among the N model summaries are identified manually. Next, SCUs of the similar sentences are produced as a pyramid model which consists of N levels labeled from 1 to N. Based on the occurrence of SCUs in the model summaries; they are ranked in the pyramid. Last, a summary is considered as good if it encloses more SCUs from the higher levels in the pyramid than the lower levels and vice versa for the poor summary (Lloret et al., 2017).

Manual evaluation and analysis of summaries consumes a lot of time and effort because it needs humans to read the summaries and also the original documents (Moratanch & Chitrakala, 2017). On the other side, most of the developed automatic evaluation methods assess the summary's content and the evaluation of readability is done almost manually (e.g. DUC and TAC conferences assess the readability of each summary manually) (Lloret et al., 2017).

5.2.2. Automatic evaluation of summaries

This subsection will explore the commonly used evaluation metrics in literature such as: 1) Precision, Recall, and F-Measure scores, 2) Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin, 2004), and 3) Basic Element (BE) (Hovy, Lin, Zhou, & Fukumoto, 2006).

Precision Score Metric: it is computed by dividing the number of sentences existing in both reference and candidate (i.e. system) summaries by the number of sentences in the candidate summary as in Eq. (3) (Moratanch & Chitrakala, 2017).

$$\text{Precision} = \frac{S_{ref} \cap S_{cand}}{S_{cand}} \tag{3}$$

Recall Score Metric: it is computed by dividing the number of sentences existing in both reference and candidate summaries by

the number of sentences in the reference summary as in Eq. (4) (Moratanch & Chitrakala, 2017).

$$\text{Recall} = \frac{S_{ref} \cap S_{cand}}{S_{ref}} \tag{4}$$

F-Measure Score Metric: it is a measure that combines recall and precision metrics as in Eq. (5) (Moratanch & Chitrakala, 2017). F-measure is the harmonic mean between precision and recall.

$$F\text{-}Measure = \frac{2(\text{Precision})(\text{Recall})}{\text{Precision} + \text{Recall}} \tag{5}$$

ROUGE Metric: it is the most commonly used tool for automatic evaluation of the automatically generated summaries (Ganesan et al., 2010). ROUGE is a set of metrics and a software package used for evaluating automatic summarization and machine translation software in NLP (Gupta & Siddiqui, 2012). It compares the computer-generated summaries against several human-created reference summaries (Lloret et al., 2017). The basic idea of ROUGE is to count the number of overlapping units between candidate (or system) summaries and the reference summaries, such as overlapped n-grams (Wang et al., 2017). ROUGE has been proven to be effective in measuring qualities of summaries and correlates well to human judgments (Sun et al., 2016). There are various ROUGE metrics as follows:

- ROUGE-1 (R1): it is based on the uni-gram measure between a candidate summary and a reference summary. ROUGE-N is an n-gram recall between a candidate summary and reference summaries.
- ROUGE-L (R-L): it is based on the longest common subsequences between a candidate summary and a reference summary.
- ROUGE-S* (R-S*): it measures the overlap ratio of skip-bigrams between a candidate summary and reference summaries.
- ROUGE-SU* (R-SU*): it extends ROUGE-S* by using skip-bigrams and using uni-gram as a counting unit. The "*" refers to the number of skip words. For example, ROUGE-SU4 permits bi-grams to consist of non-adjacent words with a maximum of four words between the two
- bi-grams words.

Although ROUGE was accepted as a standard for measuring the accuracy of a summarization model since its development, it has the disadvantage that it only matches strings between the summaries without considering the meaning in single words or series of words (n-grams). Therefore, evaluation methods to address the meaning issue have been proposed which use dependency parsing to represent the information in the candidate and reference summaries such as Basic Elements (BE) (Hovy et al., 2006), Basic Elements with Transformations for Evaluation (BEwT-E) (Tratz & Hovy, 2008), and DEPEVAL(summ) (Owczarzak, 2009) methods. In BE and BEwT-E, each sentence is divided into small content units which are called basic elements and are used to match equivalent expressions. Each basic element is a triplet of words consisting of: 1) a head, 2) a modifier or argument, and 3) the relation between the head and modifier. The main disadvantage is that these methods use many language-dependent pre-processing modules for parsing and dividing the sentences. For non-English summaries, parser resources for the summaries' languages are required (Lloret et al., 2017).

Human judgment has the disadvantage of being subjective with a wide variance on what is considered a "good" summary. This variance implies that creating an automatic evaluation and analysis method is very difficult and challenging (Moratanch & Chitrakala, 2017). In automatic evaluation, summaries generated by ATS systems are assessed by automated metrics to reduce the evaluation cost. However, human efforts are still needed by the automated evaluation metrics because they depend on the comparison of system-generated summaries with one or more human-created model summaries (Lloret et al., 2017).

6. Conclusion and future research directions

Manual text summarization is a time consuming and costly task that includes many steps. For example, the following steps are done to manually summarize a single document (Takeuchi, 2002): 1) trying to understand what the document is about, 2) trying to extract the "most important" parts from it, and 3) trying to compose a summary that satisfies the following requirements (Lloret et al., 2017):

- The summary readability and linguistic quality.
- The summary consistency and content coverage.
- The non-redundancy of the produced summary.

Due to the difficulty of manual text summarization of the huge amount of the textual content on the Internet or various archives, ATS systems have appeared as the main technology to solve this urgent and pressing issue. There are many automatic text summarizers in the literature, but they are still far away from the results of the human text summarization. It is still difficult for the computer to understand and identify the "most important" parts in the text (i.e. the importance of a text varies with its type, application domain, user preference, etc.).

There are continuous research efforts since the 1950s to overcome these difficulties and investigate new solutions. Over time, the scientific community mainly has focused on the extractive text summarization approach and has implemented the summarization methods of this approach for various types of applications such as user reviews, news articles, blogs, email messages, scientific articles, legal documents, biomedical documents, etc. In practice, extractive ATS systems produce very different summaries than the ones generated by humans. There have been some trials to propose abstractive and hybrid text summarization systems. There is still a long way to go (Moratanch & Chitrakala, 2017). Researchers dream to automate the generation of human-like summaries.

The available literature for abstractive summarization is much less than extractive text summarization. Most survey papers (Al-Saleh & Menai, 2016; Al Qassem et al., 2017; Dalal & Malik, 2013; Gambhir & Gupta, 2017; Gupta, Tiwari, & Robert, 2016; Gupta & Lehal, 2010; Meena, Jain, & Gopalani, 2014; Moratanch & Chitrakala, 2017; Shah & Desai, 2016; Tandel et al., 2016) concentrate more on the extractive text summarization because the abstractive approach is much harder and much less mature than the extractive approach. The aim of this survey is to give a comprehensive review and global overview about the different aspects of ATS. The main contributions of this survey include:

- Explaining the various classifications and the different applications of the ATS systems.
- Providing a systematic review about the ATS approaches (namely extractive, abstractive, and hybrid) and the methods that apply these approaches in the literature.
- Providing a categorization and overview of the various building blocks and techniques that have been used to design and implement the ATS systems including: 1) the text summarization operations, 2) the statistical and linguistic features, and 3) the text summarization building blocks (namely the text represen-

tation models, the linguistic analysis and processing techniques, and the soft computing techniques).
- Providing a general overview about the standard datasets, the manual evaluation criteria, and the automatic evaluation tools that are commonly used for evaluating the computer-generated summaries.
- Providing a listing and categorization of the future research directions for the ATS research community. These research directions are explained next in the rest of this section.

There are many limitations of the existing ATS systems that act as challenges and future research directions for the research community. These challenges will help the researchers to identify areas where further research is needed. Fig. 9 shows the different categories of ATS challenges that will be explained in this section.

There are some challenges related to usage of the ATS systems such as: 1) multi-document summarization, 2) user-specific summarization, and 3) applications of text summarization.

Challenges Related to Muti-Document Summarization: Multi-document summarization is a complex task and has many issues like redundancy, temporal dimension, co-reference, and sentence reordering (Gambhir & Gupta, 2017). Multi-document summarization may cause incorrect reference: one sentence may contain a proper noun and the next sentence may contain a pronoun as a reference to the proper noun. If the summarizer handles the pronoun without handling the proper noun, it will generate an improper reference (Sahoo et al., 2016).

Challenges Related to User-Specific Summarization: The key challenge here is to summarize content from a number of textual and semi-structured sources (e.g. databases and web pages) in the right way (language, format, size, and time) for a specific user (Gupta & Lehal, 2010). Due to the availability of a large amount of data in different formats and different languages, it is required to dedicate more research efforts on the multi-document, multilingual, and multimedia summaries. Also, it is required to generate summaries with a specified focus like sentiment-based, personalized summaries, etc. (Gambhir & Gupta, 2017).

Challenges Related to Applications of Text Summarization: Most of the existing systems mainly focus on certain applications like online reviews, text news, text pages, etc. (Wu et al., 2017). It is important to focus now on the most challenging applications like long text, novel, and book summarization.

Another set of challenges is related to the input and output document(s) of an ATS system such as: 1) input and output formats, 2) length of input documents, and 3) supported languages.

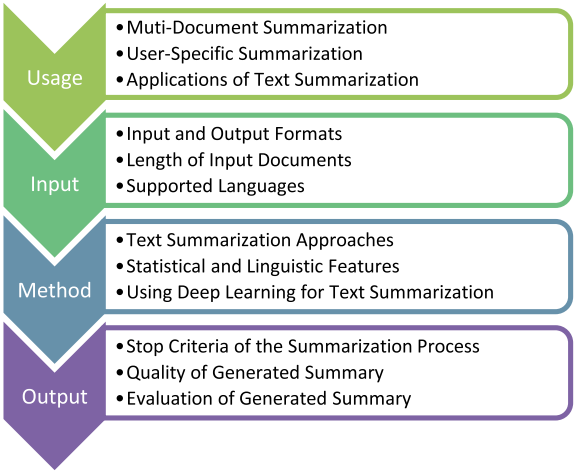

Fig. 9. Challenges for ATS systems.

Challenges Related to Input and Output Formats: Most of the ATS systems deal with textual input and output. It is required to propose new summarizers in which the input can be in the form of meetings, videos, sounds, etc. and the output in a format other than text. For example, the input might be in the form of text and the output can be represented as tables, statistics, graphics, visual rating scales, etc. ATS systems that allow visualization of the summaries will help users to get the required content in less time (Gambhir & Gupta, 2017).

Challenges Related to Length of Input Documents: Most ATS systems focus on relatively short text documents. For example, the length of a news article is shorter than that of a novel chapter (about 641 words versus 4973 words) (Wu et al., 2017). The existing ATS methods may achieve good performance in short texts, but they achieve low accuracy and efficiency when summarizing long texts (Wang et al., 2017).

Challenges Related to Supported Languages: Most ATS systems focus on the English language content. For many other languages, the quality of the current ATS systems needs to be improved. It is required to develop and improve NLP tools that are used to generate summaries for non-English languages like NER, POS tagging, syntactic and semantic parsing, etc. (Belkebir & Guessoum, 2018).

There are other challenges related to the methods and techniques of ATS systems such as: 1) text summarization approaches, 2) statistical and linguistic features, and 3) using deep learning for text summarization.

Challenges Related to Text Summarization Approaches: Most research focus on the extractive approach, it is required to focus more research efforts to propose and improve summarization systems based on the abstractive and hybrid approaches.

Challenges Related to Statistical and Linguistic Features: It is required to discover some new linguistic and statistical features for sentences and words which can semantically extract the key sentences from the source document(s) (Gambhir & Gupta, 2017). Besides, deciding the proper weights of individual features is very important because the quality of the final summary depends on it (Gupta & Lehal, 2010).

Challenges Related to Using Deep Learning for Text Summarization: In the summary generation phase, the RNN in the seq2-seq system requires a large-scale structured training data. The required training data is not always available in practical NLP applications. It is a very important research topic to build an ATS system using a small amount of training data through the combination of traditional NLP techniques such as syntactic analysis, grammar analysis, semantic analysis, etc. (Wang et al., 2017).

Finally, there are some challenges related to the output and generated summary from ATS systems such as: 1) stop criteria of the summarization process, 2) quality of generated summary, and 3) evaluation of generated summary.

Challenges Related to Stop Criteria of the Summarization Process: Humans summarize documents in an iterative process. After producing the first summary, one (or the system) should decide whether to stop or continue in the summarization process. The most common way is to set a retention rate upon which a decision is made. The retention rate is not fixed for all the texts. It should vary according to the content and type of the text. It is highly required to propose a more convincing technique to stop summarizing (Belkebir & Guessoum, 2018).

Challenges Related to the Quality of Generated Summary: It is required to achieve a good balance between readability, compression ratio, and summarization quality. It is difficult for the existing ATS systems to achieve the higher compression ratio requirement for summarizing long documents like novels and books (Wu et al., 2017). It is required to improve the summary readability by refining the initially generated summary to over-

come the semantic confusion caused by ambiguous or synonymous words (Wu et al., 2017). Without using NLP, the generated extractive summaries may suffer from lack of cohesion and semantics. Also, if texts contain multiple topics, the generated summary may not be balanced (Gupta & Lehal, 2010).

Challenges Related to Evaluation of the Generated Summary: Evaluating summaries (either automatically or manually) is a difficult task: 1) it is very challenging to define and use a good standard to evaluate whether the summaries generated from the ATS systems are good enough (Lloret et al., 2017), and 2) it is very hard to find out what an ideal (or even correct) summary is because the ATS systems can generate good summaries that are different from the human-generated summaries (Moratanch & Chitrakala, 2017). Humans are different and they may select entirely different sentences for the extractive summaries and may paraphrase the abstractive summaries in a completely different way. It is very subjective to identify a good summary. Therefore, manual evaluations may not be suitable for all types of summaries (Lloret et al., 2017). There is a need to propose new approaches and solutions for the automatic evaluation of the computer-generated summaries.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abbasi-ghalehtaki, R., Khotanlou, H., & Esmaeilpour, M. (2016). Fuzzy evolutionary cellular learning automata model for text summarization. *Swarm and Evolutionary Computation, 30*, 11–26. https://doi.org/10.1016/j.swevo.2016.03.004.

Abdolahi, M. & Zahedh, M. (2017). Sentence matrix normalization using most likely n-grams vector. Paper presented at the 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)

Afsharizadeh, M., Ebrahimpour-Komleh, H. & Bagheri, A. (2018). Query-oriented text summarization using sentence extraction technique. Paper presented at the 2018 4th International Conference on Web Research (ICWR), Tehran.

Al-Abdallah, R. Z., & Al-Taani, A. T. (2017). Arabic single-document text summarization using particle swarm optimization algorithm. *Procedia Computer Science, 117*, 30–37. https://doi.org/10.1016/j.procs.2017.10.091.

Al-Radaideh, Q. A., & Bataineh, D. Q. (2018). A hybrid approach for arabic text summarization using domain knowledge and genetic algorithms. *Cognitive Computation, 10*(4), 651–669. https://doi.org/10.1007/s12559-018-9547-z.

Al-Sabahi, K., Zhang, Z., Long, J., & Alwesabi, K. (2018). An enhanced latent semantic analysis approach for Arabic document summarization. *Arabian Journal for Science and Engineering*. https://doi.org/10.1007/s13369-018-3286-z.

Al-Saleh, A. B., & Menai, M. E. B. (2016). Automatic Arabic text summarization: A survey. *Artificial Intelligence Review, 45*(2), 203–234. https://doi.org/10.1007/s10462-015-9442-x.

Al Qassem, L. M., Wang, D., Al Mahmoud, Z., Barada, H., Al-Rubaie, A., & Almoosa, N. I. (2017). Automatic Arabic summarization: A survey of methodologies and systems. *Procedia Computer Science, 117*, 10–18. https://doi.org/10.1016/j.procs.2017.10.088.

Alami, N., El Adlouni, Y., En-nahnahi, N. & Meknassi, M. (2018). Using statistical and semantic analysis for Arabic text summarization. Paper presented at the ITCS 2017: International Conference on Information Technology and Communication Systems, Khouribga, Morocco.

Alami, N., Meknassi, M., & En-nahnahi, N. (2019). Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning. *Expert Systems with Applications, 123*, 195–211. https://doi.org/10.1016/j.eswa.2019.01.037.

Alampalli Ramu, N., Bandarupalli, M. S., Nekkanti, M. S. S. & Ramesh, G. (2020). Summarization of research publications using automatic extraction. Paper presented at the ICICI 2019: International Conference on Intelligent Data Communication Technologies and Internet of Things, Coimbatore, India.

Alguliyev, R. M., Aliguliyev, R. M., Isazade, N. R., Abdi, A., & Idris, N. (2019). COSUM: Text summarization based on clustering and optimization. *Expert Systems, 36*(1). https://doi.org/10.1111/exsy.12340 e12340.

Anand, D., & Wagh, R. (2019). Effective deep learning approaches for summarization of legal texts. *Journal of King Saud University – Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2019.11.015.

Baralis, E., Cagliero, L., Mahoto, N., & Fiori, A. (2013). GraphSum: Discovering correlations among multiple terms for graph-based summarization. *Information Sciences, 249*, 96–109. https://doi.org/10.1016/j.ins.2013.06.046.

Belkebir, R. & Guessoum, A. (2018). TALAA-ATSF: A global operation-based Arabic text summarization framework. In K. Shaalan, A. E. Hassanien & F. Tolba (Eds.), Intelligent natural language processing: Trends and applications (pp. 435–459). Springer International Publishing.

Bhargava, R. & Sharma, Y. (2017). MSATS: Multilingual sentiment analysis via text summarization. Paper presented at the 2017 7th International Conference on Cloud Computing, Data Science & Engineering – Confluence.

Bhat, I. K., Mohd, M. & Hashmy, R. (2018). SumItUp: A hybrid single-document text summarizer. In M. Pant, K. Ray, T. K. Sharma, S. Rawat & A. Bandyopadhyay (Eds.), Soft computing: Theories and applications: Proceedings of SoCTA 2016, Volume 1 (pp. 619–634). Singapore: Springer Singapore.

Bhattacharya, P., Hiware, K., Rajgaria, S., Pochhi, N., Ghosh, K. & Ghosh, S. (2019). A comparative study of summarization algorithms applied to legal case judgments. Paper presented at the Advances in Information Retrieval, Cham.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research, 3*, 993–1022.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems, 30*(1), 107–117. https://doi.org/10.1016/S0169-7552(98)00110-X.

Cai, T., Shen, M., Peng, H., Jiang, L. & Dai, Q. (2019). Improving transformer with sequential context representations for abstractive text summarization. Paper presented at the Natural Language Processing and Chinese Computing, Cham.

Carenini, G., Ng, R. T., & Zhou, X. (2007). Summarizing email conversations with clue words. Paper presented at the Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada.

Cavnar, W. (1994). Using an n-gram-based document representation with a vector processing retrieval model. Paper presented at the 3rd Text Retrieval Conference, Gaithersburg, Maryland, US.

Chakraborty, R., Bhavsar, M., Dandapat, S. K., & Chandra, J. (2019). Tweet summarization of news articles: An objective ordering-based perspective. *IEEE Transactions on Computational Social Systems, 6*(4), 761–777. https://doi.org/10.1109/TCSS.2019.2926144.

Chatterjee, N., Mittal, A. & Goyal, S. (2012). Single document extractive text summarization using Genetic Algorithms. Paper presented at the 2012 Third International Conference on Emerging Applications of Information Technology.

Chen, L. & Nguyen, M. L. (2019). Sentence selective neural extractive summarization with reinforcement learning. Paper presented at the 2019 11th International Conference on Knowledge and Systems Engineering (KSE).

Cheng, J. & Lapata, M. (2016). Neural summarization by extracting sentences and words. Paper presented at the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany.

Chitrakala, S., Moratanch, N., Ramya, B., Revanth Raaj, C. G. & Divya, B. (2018). Concept-based extractive text summarization using graph modelling and weighted iterative ranking. In N. R. Shetty, L. M. Patnaik, N. H. Prasad & N. Nalini (Eds.), Emerging research in computing, information, communication and applications: ERCICA 2016 (pp. 149–160). Singapore: Springer Singapore.

Chopra, S., Auli, M. & M. Rush, A. (2016). Abstractive sentence summarization with attentive recurrent neural networks. Paper presented at the NAACL-HLT 2016, San Diego, California.

Clarke, J., Goldwasser, D., Chang, M. -W. & Roth, D. (2010). Driving semantic parsing from the world's response. Paper presented at the Proceedings of the Fourteenth Conference on Computational Natural Language Learning, Uppsala, Sweden.

Cohan, A., & Goharian, N. (2018). Scientific document summarization via citation contextualization and scientific discourse. *International Journal on Digital Libraries, 19*(2), 287–303. https://doi.org/10.1007/s00799-017-0216-8.

Cunningham, H., Maynard, D., Bontcheva, K. & Tablan, V. (2002). GATE: An architecture for development of robust HLT applications. Paper presented at the the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia

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

Dalal, V. & Malik, L. (2018). Semantic graph based automatic text summarization for Hindi documents using particle swarm optimization. In S. C. Satapathy & A. Joshi (Eds.), Information and communication technology for intelligent systems (ICTIS 2017) (Vol. 2, pp. 284–289). Springer International Publishing.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391–407. https://doi.org/10.1002/(sici)1097-4571(199009)41:6<391::aid-asi1>3.0.co;2-9.

Dernoncourt, F., Ghassemi, M. & Chang, W. (2018). A repository of corpora for summarization, Miyazaki, Japan.

dos Santos, F. F., Domingues, M. A., Sundermann, C. V., de Carvalho, V. O., Moura, M. F., & Rezende, S. O. (2018). Latent association rule cluster based model to extract topics for classification and recommendation applications. *Expert Systems with Applications, 112*, 34–60. https://doi.org/10.1016/j.eswa.2018.06.021.

Dutta, M., Das, A. K., Mallick, C., Sarkar, A. & Das, A. K. (2019). A graph based approach on extractive summarization. Paper presented at the International Conference on Emerging Technologies in Data Mining and Information Security (IEMIS 2018) Kolkata, India.

Dutta, S., Chandra, V., Mehra, K., Ghatak, S., Das, A. K. & Ghosh, S. (2019). Summarizing microblogs during emergency events: A comparison of extractive summarization algorithms. Paper presented at the International Conference on Emerging Technologies in Data Mining and Information Security (IEMIS 2018), Kolkata, India.

El-Haj, M., Kruschwitz, U., & Fox, C. (2015). Creating language resources for under-resourced languages: Methodologies, and experiments with Arabic. *Language Resources and Evaluation, 49*(3), 549–580. https://doi.org/10.1007/s10579-014-9274-3.

Embar, V. R., Deshpande, S. R., Vaishnavi, A. K., Jain, V. & Kallimani, J. S. (2013). sArAmsha – A Kannada abstractive summarizer. Paper presented at the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Ercan, G., & Cicekli, I. (2007). Using lexical chains for keyword extraction. *Information Processing & Management, 43*(6), 1705–1714. https://doi.org/10.1016/j.ipm.2007.01.015.

Erkan, G., & Radev, D. R. (2004). LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research, 22*, 457–479.

Ermakova, L., Cossu, J. V., & Mothe, J. (2019). A survey on evaluation of summarization methods. *Information Processing & Management, 56*(5), 1794–1814. https://doi.org/10.1016/j.ipm.2019.04.001.

Fang, C., Mu, D., Deng, Z., & Wu, Z. (2017). Word-sentence co-ranking for automatic extractive text summarization. *Expert Systems with Applications, 72*, 189–195. https://doi.org/10.1016/j.eswa.2016.12.021.

Finegan-Dollak, C. (2018). Selecting and generating computational meaning representations for short texts. (PhD thesis). University of Michigan.

Gambhir, M., & Gupta, V. (2017). Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review, 47*(1), 1–66. https://doi.org/10.1007/s10462-016-9475-9.

Ganesan, K., Zhai, C., & Han, J. (2010). Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions. Paper presented at the Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China.

García-Hernández, R. A. & Ledeneva, Y. (2013). Single extractive text summarization based on a genetic algorithm. Paper presented at the Pattern Recognition, Berlin, Heidelberg.

Gehrmann, S., Deng, Y. & Rush, A. (2018). Bottom-up abstractive summarization, Brussels, Belgium.

Genest, P.-E. & Lapalme, G. (2012). Fully abstractive approach to guided summarization. Paper presented at the Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers – Volume 2, Jeju Island, Korea.

Goldstein, J., Kantrowitz, M., Mittal, V. & Carbonell, J. (1999). Summarizing text documents: Sentence selection and evaluation metrics. Paper presented at the Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, California, USA.

Gupta, P., Tiwari, R., & Robert, N. (2016). Sentiment analysis and text summarization of online reviews: A survey. Paper presented at the 2016 International Conference on Communication and Signal Processing (ICCSP).

Gupta, S., & Gupta, S. K. (2019). Abstractive summarization: An overview of the state of the art. *Expert Systems with Applications, 121*, 49–65. https://doi.org/10.1016/j.eswa.2018.12.011.

Gupta, V., Bansal, N. & Sharma, A. (2019). Text summarization for big data: A comprehensive survey. Paper presented at the International Conference on Innovative Computing and Communications, Singapore.

Gupta, V., & Lehal, G. S. (2010). A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence, 2*(3), 258–268.

Gupta, V. K. & Siddiqui, T. J. (2012). Multi-document summarization using sentence clustering. Paper presented at the 2012 4th international conference on intelligent human computer interaction (IHCI).

Hahn, U., & Mani, I. (2000). The challenges of automatic summarization. *Computer, 33*(11), 29–36. https://doi.org/10.1109/2.881692.

Harris, Z. S. (1954). Distributional structure. *WORD, 10*(2–3), 146–162. https://doi.org/10.1080/00437956.1954.11659520.

Hasler, L. (2007). From extracts to abstracts: Human summary production operations for computer-aided summarisation. (PhD), University of Wolverhampton.

Hasler, L., Orasan, C. & Mitkov, R. (2003). Building better corpora for summarisation. Paper presented at the Corpus Linguistics 2003, Lancaster, UK.

Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M. & Blunsom, P. (2015). Teaching machines to read and comprehend. Paper presented at the Proceedings of the 28th International Conference on Neural Information Processing Systems – Volume 1, Montreal, Canada.

Hofmann, T. (1999). Probabilistic latent semantic indexing. Paper presented at the Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, California, USA.

Hou, L., Hu, P. & Bei, C. (2017). Abstractive document summarization via neural model with joint attention. Paper presented at the Natural Language Processing and Chinese Computing, Dalian, China.

Hovy, E., Lin, C.-Y., Zhou, L. & Fukumoto, J. (2006). Automated summarization evaluation with basic elements. Paper presented at the the 5th Conference on Language Resources and Evaluation.

Hu, B., Chen, Q. & Zhu, F. (2015). LCSTS: A large scale Chinese short text summarization dataset. CoRR abs/1506.05865.

Huang, L., He, Y., Wei, F. & Li, W. (2010). Modeling document summarization as multi-objective optimization. Paper presented at the 2010 Third International Symposium on Intelligent Information Technology and Security Informatics.

Ibrahim, A., Elghazaly, T. & Gheith, M. (2013). A novel Arabic text summarization model based on rhetorical structure theory and vector space model. International Journal of Computational Linguistics and Natural Language Processing, 2(8).

Ibrahim, D. (2016). An overview of soft computing. *Procedia Computer Science, 102*, 34–38. https://doi.org/10.1016/j.procs.2016.09.366.

Indurkhya, N. & Damerau, F. J. (2010). Handbook of Natural language processing (2nd ed.). Chapman & Hall/CRC.

J Kurisinkel, L., Zhang, Y. & Varma, V. (2017). Abstractive multi-document summarization by partial tree extraction, recombination and linearization. Paper presented at the Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Taipei, Taiwan.

Jacquenet, F., Bernard, M. & Largeron, C. (2019). Meeting summarization, a challenge for deep learning. Paper presented at the Advances in Computational Intelligence, Cham.

Jafari, M., Wang, J., Qin, Y., Gheisari, M., Shahabi, A. S. & Tao, X. (2016). Automatic text summarization using fuzzy inference. Paper presented at the 2016 22nd International Conference on Automation and Computing (ICAC).

Jaradat, Y. A. & Al-Taani, A. T. (2016). Hybrid-based Arabic single-document text summarization approach using genatic algorithm. Paper presented at the 2016 7th International Conference on Information and Communication Systems (ICICS).

Jiang, X. -J., Mao, X. -L., Feng, B. -S., Wei, X., Bian, B. -B. & Huang, H. (2019). HSDS: An abstractive model for automatic survey generation. Paper presented at the Database Systems for Advanced Applications, Cham

Jing, H. (2002). Using hidden Markov modeling to decompose human-written summaries. *Computational Linguistics, 28*(4), 527–543. https://doi.org/10.1162/089120102762671972.

John, A. & Wilscy, M. (2013). Random forest classifier based multi-document summarization system. Paper presented at the 2013 IEEE Recent Advances in Intelligent Computational Systems (RAICS)

Joshi, M., Wang, H. & McClean, S. (2018). Dense semantic graph and its application in single document summarisation. In C. Lai, A. Giuliani & G. Semeraro (Eds.), Emerging ideas on information filtering and retrieval: DART 2013: Revised and invited papers (pp. 55–67). Springer International Publishing.

Jurafsky, D. & Martin, J. H. (2017). Speech and language processing (3rd ed.).

Kanapala, A., Pal, S., & Pamula, R. (2019). Text summarization from legal documents: A survey. *Artificial Intelligence Review, 51*(3), 371–402. https://doi.org/10.1007/s10462-017-9566-2.

Kavila, S. D., Puli, V., Prasada Raju, G. S. V. & Bandaru, R. (2013). An automatic legal document summarization and search using hybrid system, Berlin, Heidelberg.

Kazantseva, A., & Szpakowicz, S. (2010). Summarizing short stories. *Computational Linguistics, 36*(1), 71–109. https://doi.org/10.1162/coli.2010.36.1.36102.

Khan, A., Salim, N. & Farman, H. (2016). Clustered genetic semantic graph approach for multi-document abstractive summarization. Paper presented at the 2016 International Conference on Intelligent Systems Engineering (ICISE)

Khan, A., Salim, N., Farman, H., Khan, M., Jan, B., Ahmad, A., ... Paul, A. (2018). Abstractive text summarization based on improved semantic graph approach. *International Journal of Parallel Programming, 46*(5), 992–1016. https://doi.org/10.1007/s10766-018-0560-3.

Khan, A., Salim, N., & Jaya Kumar, Y. (2015). A framework for multi-document abstractive summarization based on semantic role labelling. *Applied Soft Computing, 30*, 737–747. https://doi.org/10.1016/j.asoc.2015.01.070.

Kim, H. D., Park, D. H., Lu, Y. & Zhai, C. (2012). Enriching text representation with frequent pattern mining for probabilistic topic modeling. Proceedings of the American Society for Information Science and Technology, 49(1), 1–10. doi: 10.1002/meet.14504901209.

Kirmani, M., Manzoor Hakak, N., Mohd, M. & Mohd, M. (2019). Hybrid text summarization: A survey. Paper presented at the Soft Computing: Theories and Applications, Singapore

Ko, Y., & Seo, J. (2008). An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. *Pattern Recognition Letters, 29*(9), 1366–1371. https://doi.org/10.1016/j.patrec.2008.02.008.

Kobayashi, H., Noguchi, M. & Yatsuka, T. (2015). Summarization based on embedding distributions. Paper presented at the Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal

Kouris, P., Alexandridis, G. & Stafylopatis, A. (2019). Abstractive text summarization based on deep learning and semantic content generalization. Paper presented at the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy.

Krishnakumari, K. & Sivasankar, E. (2018). Scalable aspect-based summarization in the hadoop environment. In V. B. Aggarwal, V. Bhatnagar & D. K. Mishra (Eds.), Big data analytics: Proceedings of CSI 2015 (pp. 439–449). Singapore: Springer Singapore.

Kumar, A., & Sharma, A. (2019). Systematic literature review of fuzzy logic based text summarization. *Iranian Journal of Fuzzy Systems, 16*(5), 45–59. https://doi.org/10.22111/ijfs.2019.4906.

Kurup, L. & Narvekar, M. (2020). A roadmap to realization approaches in natural language generation, Singapore.

Le, H. T. & Le, T. M. (2013). An approach to abstractive text summarization. Paper presented at the 2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR)

Levelt, P., & Caramazza, A. (2007). *The Oxford handbook of psycholinguistics*. USA: Oxford University Press.

Li, H., Zhu, J., Ma, C., Zhang, J., & Zong, C. (2018). Read, watch, listen and summarize: Multi-modal summarization for asynchronous text, image, audio and video. *IEEE Transactions on Knowledge and Data Engineering, 1–1*. https://doi.org/10.1109/TKDE.2018.2848260.

Li, X., Sun, M. & Li, P. (2019). Multi-agent discussion mechanism for natural language generation. Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence.
Lin, C. -Y. (2004). ROUGE: A package for automatic evaluation of summaries. Paper presented at the Workshop on Text Summarization Branches Out, Barcelona, Spain.
Lin, H. & Ng, V. (2019). Abstractive summarization: A survey of the state of the art. Paper presented at the The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19).
Linhares Pontes, E., Huet, S., Torres-Moreno, J.-M., & Linhares, A. C. (2020). Compressive approaches for cross-language multi-document summarization. *Data & Knowledge Engineering, 125*. https://doi.org/10.1016/j.datak.2019.101763 101763.
Lins, R. D., Mello, R. F. & Simske, S. (2019). DocEng'19 competition on extractive text summarization. Paper presented at the Proceedings of the ACM Symposium on Document Engineering 2019, Berlin, Germany. https://doi.org/10.1145/3342558.3351874
Lins, R. D., Oliveira, H., Cabral, L., Batista, J., Tenorio, B., Ferreira, R., . . . Simske, S. J. (2019). The CNN-corpus: A large textual corpus for single-document extractive summarization. Paper presented at the Proceedings of the ACM Symposium on Document Engineering 2019, Berlin, Germany. https://doi.org/10.1145/3342558.3345388
Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, Ł. & Shazeer, N. (2018). Generating wikipedia by summarizing long sequences. Paper presented at the ICLR 2018. https://arxiv.org/pdf/1801.10198.pdf (Last Accessed: 26/4/2020)
Lloret, E. & Palomar, M. (2009). A gradual combination of features for building automatic summarisation systems. Paper presented at the International Conference on Text, Speech and Dialogue, Pilsen, Czech Republic.
Lloret, E., Plaza, L., & Aker, A. (2017). The challenging task of summary evaluation: An overview. *Language Resources and Evaluation*. https://doi.org/10.1007/s10579-017-9399-2.
Lloret, E., Romá-Ferri, M. T. & Palomar, M. (2011). COMPENDIUM: A text summarization system for generating abstracts of research papers. Paper presented at the natural language processing and information systems, Berlin, Heidelberg
Lloret, E., Romá-Ferri, M. T., & Palomar, M. (2013). COMPENDIUM: A text summarization system for generating abstracts of research papers. *Data & Knowledge Engineering, 88*, 164–175. https://doi.org/10.1016/j.datak.2013.08.005.
Lovinger, J., Valova, I., & Clough, C. (2019). Gist: General integrated summarization of text and reviews. *Soft Computing, 23*(5), 1589–1601. https://doi.org/10.1007/s00500-017-2882-2.
Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development, 2*(2), 159–165. https://doi.org/10.1147/rd.22.0159.
Mahajani, A., Pandya, V., Maria, I. & Sharma, D. (2019). A comprehensive survey on extractive and abstractive techniques for text summarization. Paper presented at the ambient communications and computer systems, Singapore
Maiti, S., Garain, U., Dhar, A., & De, S. (2015). A novel method for performance evaluation of text chunking. *Language Resources and Evaluation, 49*(1), 215–226. https://doi.org/10.1007/s10579-013-9250-3.
Mallick, C., Das, A. K., Dutta, M., Das, A. K. & Sarkar, A. (2019). Graph-based text summarization using modified TextRank, Singapore
Mandal, S., Singh, G. K. & Pal, A. (2019). PSO-Based Text Summarization Approach Using Sentiment Analysis. Paper presented at the computing, communication and signal processing, Singapore
Mani, I. (2001). Automatic summarization (Vol. 3). John Benjamins Publishing Company.
Mann William, C., & Thompson Sandra, A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text – Interdisciplinary Journal for the Study of Discourse, 8*, 243.
Manning, C. D., Raghavan, P. & Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press.
Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J. & McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. Paper presented at the 52nd annual meeting of the association for computational linguistics: System demonstrations, Baltimore, Maryland, USA.
Mao, X., Yang, H., Huang, S., Liu, Y., & Li, R. (2019). Extractive summarization using supervised and unsupervised learning. *Expert Systems with Applications, 133*, 173–181. https://doi.org/10.1016/j.eswa.2019.05.011.
Marques, J. M. C., Cozman, F. G. & Santos, I. H. F. d. (2019). Automatic summarization of technical documents in the oil and gas industry. Paper presented at the 2019 8th Brazilian conference on intelligent systems (BRACIS).
Mart, #237, Abadi, n., Barham, P., Chen, J., Chen, Z., . . . Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. Paper presented at the proceedings of the 12th USENIX conference on operating systems design and implementation, Savannah, GA, USA.
Mary, A. J. J., & Arockiam, L. (2017). ASFuL: Aspect based sentiment summarization using fuzzy logic. Paper presented at the 2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET).
Mashechkin, I. V., Petrovskiy, M. I., Popov, D. S., & Tsarev, D. V. (2011). Automatic text summarization using latent semantic analysis. *Programming and Computer Software, 37*(6), 299–305. https://doi.org/10.1134/s0361768811060041.
Maybury, M. T. (1995). Generating summaries from event data. *Information Processing & Management, 31*(5), 735–751. https://doi.org/10.1016/0306-4573(95)00025-C.
McKeown, K. R., Barzilay, R., Evans, D., Hatzivassiloglou, V., Klavans, J. L., Nenkova, A., . . . Sigelman, S. (2002). Tracking and summarizing news on a daily basis with Columbia's Newsblaster. Paper presented at the Proceedings of the second international conference on Human Language Technology Research, San Diego, California.
Meena, Y. K., & Gopalani, D. (2015). Evolutionary algorithms for extractive automatic text summarization. *Procedia Computer Science, 48*, 244–249. https://doi.org/10.1016/j.procs.2015.04.177.
Meena, Y. K., Jain, A. & Gopalani, D. (2014). Survey on graph and cluster based approaches in multi-document text summarization. Paper presented at the International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)
Mehta, P., & Majumder, P. (2018). Effective aggregation of various summarization techniques. *Information Processing & Management, 54*(2), 145–158. https://doi.org/10.1016/j.ipm.2017.11.002.
Melucci, M. (2009). Vector-space model. In L. Liu & M. T. ÖZsu (Eds.), Encyclopedia of database systems (pp. 3259–3263). Boston, MA: Springer US.
Menéndez, H. D., Plaza, L. & Camacho, D. (2014). Combining graph connectivity and genetic clustering to improve biomedical summarization. Paper presented at the 2014 IEEE Congress on Evolutionary Computation (CEC).
Merchant, K., & Pande, Y. (2018). NLP based latent semantic analysis for legal text summarization. Paper presented at the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI).
Miao, W., Zhang, G., Bai, Y. & Cai, D. (2019). Improving accuracy of key information acquisition for social media text summarization. Paper presented at the 2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS).
Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. Paper presented at the Proceedings of the ACL 2004 on Interactive poster and demonstration sessions, Barcelona, Spain.
Mihalcea, R., & Ceylan, H. (2007). Explorations in Automatic book summarization. Paper presented at the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague.
Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into texts. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain.
Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). Efficient estimation of word representations in vector space. https://arxiv.org/pdf/1301.3781.pdf (Last Accessed: 26/4/2020)
Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM, 38*, 39–41.
Mingzhen, C. & Yu, S. (2009). Summarization of text clustering based vector space model. Paper presented at the 2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design.
Mirani, T. B. & Sasi, S. (2017). Two-level text summarization from online news sources with sentiment analysis. Paper presented at the 2017 International Conference on Networks & Advances in Computational Technologies (NetACT).
Mogren, O., Kageback, M. & Dubhashi, D. (2015). Extractive summarization by aggregating multiple similarities. Paper presented at the Recent Advances in Natural Language Processing, Hissar, Bulgaria.
Mohamed, M., & Oussalah, M. (2019). SRL-ESA-TextSum: A text summarization approach based on semantic role labeling and explicit semantic analysis. *Information Processing & Management, 56*(4), 1356–1372. https://doi.org/10.1016/j.ipm.2019.04.003.
Mohammad, S., Dorr, B., Egan, M., Hassan, A., Muthukrishan, P., Qazvinian, V., . . . Zajic, D. (2009). Using citations to generate surveys of scientific paradigms. Paper presented at the Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, Colorado.
Mohan, M. J., Sunitha, C., Ganesh, A., & Jaya, A. (2016). A study on ontology based abstractive summarization. *Procedia Computer Science, 87*, 32–37. https://doi.org/10.1016/j.procs.2016.05.122.
Mohd, M., Jan, R., & Shah, M. (2020). Text document summarization using word embedding. *Expert Systems with Applications, 143*, 112958. https://doi.org/10.1016/j.eswa.2019.112958.
Morales, L. P., D, A., #237, Esteban, a., Gerv, P., & #225. (2008). Concept-graph based biomedical automatic summarization using ontologies. Paper presented at the Proceedings of the 3rd Textgraphs Workshop on Graph-Based Algorithms for Natural Language Processing, Manchester, United Kingdom.
Moratanch, N. & Chitrakala, S. (2016). A survey on abstractive text summarization. Paper presented at the 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT).
Moratanch, N. & Chitrakala, S. (2017). A Survey on Extractive Text Summarization. Paper presented at the 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai.
Mosa, M. A., Anwar, A. S., & Hamouda, A. (2019). A survey of multiple types of text summarization with their satellite contents based on swarm intelligence optimization algorithms. *Knowledge-Based Systems, 163*, 518–532. https://doi.org/10.1016/j.knosys.2018.09.008.
Muresan, S., Tzoukermann, E. & Klavans, J. L. (2001). Combining linguistic and machine learning techniques for email summarization. Paper presented at the Proceedings of the 2001 workshop on Computational Natural Language Learning – Volume 7, Toulouse, France.

Mutlu, B., Sezer, E. A., & Akcayol, M. A. (2019). Multi-document extractive text summarization: A comparative assessment on features. *Knowledge-Based Systems, 183*. https://doi.org/10.1016/j.knosys.2019.07.019 104848.

Nallapati, R., Zhai, F., & Zhou, B. (2017). SummaRuNNer: A of recurrent neural network based sequence model for extractive summarization documents. Paper presented at the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, California, USA.

Nallapati, R., Zhou, B., santos, C. N. d., Gulcehre, C. & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence RNNs and beyond.

Nasar, Z., Jaffry, S. W., & Malik, M. K. (2019). Textual keyword extraction and summarization: State-of-the-art. *Information Processing & Management, 56*(6). https://doi.org/10.1016/j.ipm.2019.102088.

Nasr Azadani, M., Ghadiri, N., & Davoodijam, E. (2018). Graph-based biomedical text summarization: An itemset mining and sentence clustering approach. *Journal of Biomedical Informatics, 84*, 42–58. https://doi.org/10.1016/j.jbi.2018.06.005.

Nazari, N., & Mahdavi, M. A. (2019). A survey on automatic text summarization. *Journal of AI and Data Mining, 7*(1), 121–135. https://doi.org/10.22044/jadm.2018.6139.1726.

Nenkova, A. & McKeown, K. (2012). A survey of text summarization techniques. In C. C. Aggarwal & C. Zhai (Eds.), Mining text data (pp. 43–76). Boston, MA: Springer US.

Nenkova, A. & Passonneau, R. (2004). Evaluating content selection in summarization: The pyramid method, Boston, Massachusetts, USA.

Nenkova, A., Passonneau, R., & McKeown, K. (2007). The Pyramid Method: Incorporating human content selection variation in summarization evaluation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 4*(2), 4. https://doi.org/10.1145/1233912.1233913.

Neri Mendoza, V., Ledeneva, Y. & García-Hernández, R. A. (2019). Abstractive multi-document text summarization using a genetic algorithm. Paper presented at the Pattern Recognition, Cham.

Okumura, N. & Miura, T. (2015). Automatic labelling of documents based on ontology. Paper presented at the 2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM).

Owczarzak, K. (2009). DEPEVAL(summ): Dependency-based evaluation for automatic summaries. Paper presented at the Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 – Volume 1, Suntec, Singapore.

Oya, T., Mehdad, Y., Carenini, G. & Ng, R. (2014). A template-based abstractive meeting summarization: Leveraging summary and source text relationships. Paper presented at the Proceedings of the 8th International Natural Language Generation Conference (INLG), Philadelphia, Pennsylvania, USA.

Patel, D., Shah, S., & Chhinkaniwala, H. (2019). Fuzzy logic based multi document summarization with improved sentence scoring and redundancy removal technique. *Expert Systems with Applications, 134*, 167–177. https://doi.org/10.1016/j.eswa.2019.05.045.

Patil, A. P., Dalmia, S., Ansari, S. A. A., Aul, T. & Bhatnagar, V. (2014). Automatic text summarizer. Paper presented at the 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Priya, V., & Umamaheswari, K. (2019). Enhanced continuous and discrete multi objective particle swarm optimization for text summarization. *Cluster Computing, 22*(1), 229–240. https://doi.org/10.1007/s10586-018-2674-1.

Qassem, L. A., Wang, D., Barada, H., Al-Rubaie, A. & Almoosa, N. (2019). Automatic Arabic text summarization based on fuzzy logic. Paper presented at the Proceedings of the 3rd International Conference on Natural Language and Speech Processing, Trento, Italy.

Radev, D., Teufel, S., Saggion, H., Lam, W., Blitzer, J., Qi, H., ... Drabek, E. (2003). Evaluation challenges in large-scale multi-document summarization: The MEAD project.

Radev, D. R., Blair-Goldensohn, S. & Zhang, Z. (2001). Experiments in single and multi-document summarization using MEAD. Paper presented at the First Document Understanding Conference, New Orleans, LA.

Radev, D. R., Hovy, E. & McKeown, K. (2002). Introduction to the special issue on summarization. Computational Linguistics, 28(4), 399-408. doi: 10.1162/089120102762671927.

Radev, D. R., Jing, H., Styś, M., & Tam, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management, 40*(6), 919–938. https://doi.org/10.1016/j.ipm.2003.10.006.

Rahman, A., Rafiq, F. M., Saha, R., Rafian, R. & Arif, H. (2019). Bengali text summarization using TextRank, fuzzy C-Means and aggregate scoring methods. Paper presented at the 2019 IEEE Region 10 Symposium (TENSYMP).

Ranjitha, N. S. & Kallimani, J. S. (2017). Abstractive multi-document summarization. Paper presented at the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Rao, K. K. & Svp Raju, G. (2011). An overview on soft computing techniques. Paper presented at the High Performance Architecture and Grid Computing, Berlin, Heidelberg.

Reeve, L., Han, H. & Brooks, A. D. (2006). BioChain: Lexical chaining methods for biomedical text summarization. Paper presented at the Proceedings of the 2006 ACM symposium on Applied computing, Dijon, France.

Reeve, L. H., Han, H., & Brooks, A. D. (2007). The use of domain-specific concepts in biomedical text summarization. *Information Processing & Management, 43*(6), 1765–1776. https://doi.org/10.1016/j.ipm.2007.01.026.

Rehurek, R. & Sojka, P. (2010). Software framework for topic modelling with large corpora. Paper presented at the LREC 2010 Workshop on New Challenges for NLP Framework. https://radimrehurek.com/gensim/index.html.

Roul, R. K., & Arora, K. (2019). A nifty review to text summarization-based recommendation system for electronic products. *Soft Computing, 23*(24), 13183–13204. https://doi.org/10.1007/s00500-019-03861-3.

Rudra, K., Goyal, P., Ganguly, N., Imran, M., & Mitra, P. (2019). Summarizing situational tweets in crisis scenarios: An extractive-abstractive approach. *IEEE Transactions on Computational Social Systems, 6*(5), 981–993. https://doi.org/10.1109/TCSS.2019.2937899.

Sahba, R., Ebadi, N., Jamshidi, M. & Rad, P. (2018). Automatic text summarization using customizable fuzzy features and attention on the context and vocabulary. Paper presented at the 2018 World Automation Congress (WAC).

Sahni, A. & Palwe, S. (2017). Topic Modeling On Online News Extraction. Paper presented at the Intelligent Computing and Information and Communication, Singapore.

Sahoo, D., Balabantaray, R., Phukon, M. & Saikia, S. (2016). Aspect based multi-document summarization. Paper presented at the 2016 International Conference on Computing, Communication and Automation (ICCCA).

Sahoo, D., Bhoi, A., & Balabantaray, R. C. (2018). Hybrid approach to abstractive summarization. *Procedia Computer Science, 132*, 1228–1237. https://doi.org/10.1016/j.procs.2018.05.038.

Sakhare, D. Y., Kumar, R. & Janmeda, S. (2018). Development of embedded platform for Sanskrit grammar-based document summarization. In S. S. Agrawal, A. Devi, R. Wason & P. Bansal (Eds.), Speech and language processing for human-machine communications: Proceedings of CSI 2015 (pp. 41–50). Singapore: Springer Singapore.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 18*(11), 613–620. https://doi.org/10.1145/361219.361220.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., & Pérez, C. J. (2018). Extractive multi-document text summarization using a multi-objective artificial bee colony optimization approach. *Knowledge-Based Systems, 159*, 1–8. https://doi.org/10.1016/j.knosys.2017.11.029.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., & Pérez, C. J. (2020a). A decomposition-based multi-objective optimization approach for extractive multi-document text summarization. *Applied Soft Computing, 91*, 106231. https://doi.org/10.1016/j.asoc.2020.106231.

Sanchez-Gomez, J. M., Vega-Rodríguez, M. A., & Pérez, C. J. (2020b). Experimental analysis of multiple criteria for extractive multi-document text summarization. *Expert Systems with Applications, 140*. https://doi.org/10.1016/j.eswa.2019.112904 112904.

Sankarasubramaniam, Y., Ramanathan, K., & Ghosh, S. (2014). Text summarization using Wikipedia. *Information Processing & Management, 50*(3), 443–461. https://doi.org/10.1016/j.ipm.2014.02.001.

Sarracén, G. L. D. l. P. & Rosso, P. (2018). Automatic text summarization based on betweenness centrality. Paper presented at the 5th Spanish Conference on Information Retrieval, Zaragoza, Spain.

See, A., Liu, P. J. & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. Paper presented at the the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada.

Sethi, P., Sonawane, S., Khanwalker, S. & Keskar, R. B. (2017). Automatic text summarization of news articles. Paper presented at the 2017 International Conference on Big Data, IoT and Data Science (BID).

Shah, P. & Desai, N. P. (2016). A survey of automatic text summarization techniques for Indian and foreign languages. Paper presented at the 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT).

Sheth, M., Popat, S. & Vyas, T. (2018). Word sense disambiguation for Indian languages, Singapore.

Shetty, K. & Kallimani, J. S. (2017). Automatic extractive text summarization using K-means clustering. Paper presented at the 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT).

Singh, K. N., Devi, H. M., & Mahanta, A. K. (2017). Document representation techniques and their effect on the document Clustering and Classification: A review. *International Journal of Advanced Research in Computer Science, 8*(5).

Suleiman, D. & Awajan, A. A. (2019). Deep learning based extractive text summarization: Approaches, datasets and evaluation measures. Paper presented at the 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS).

Sun, R., Wang, Z., Ren, Y. & Ji, D. (2016). Query-biased multi-document abstractive summarization via submodular maximization using event guidance. Paper presented at the Web-Age Information Management, Nanchang, China.

Takeuchi, K. (2002). A study on operations used in text summarization. (PhD thesis), Nara Institute of Science and Technology.

Tandel, A., Modi, B., Gupta, P., Wagle, S. & Khedkar, S. (2016). Multi-document text summarization – a survey. Paper presented at the 2016 International Conference on Data Mining and Advanced Computing (SAPIENCE).

Tandel, J., Mistree, K. & Shah, P. (2019). A review on neural network based abstractive text summarization models. Paper presented at the 2019 IEEE 5th International Conference for Convergence in Technology (I2CT).

Teufel, S., & Moens, M. (2002). Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics, 28*(4), 409–445. https://doi.org/10.1162/089120102762671936.

Tratz, S. & Hovy, E. (2008). Summarization evaluation using transformed basic elements. Paper presented at the Text Analytics Conference (TAC-08), NIST, Gaithersburg, MD.

Tuarob, S., Bhatia, S., Mitra, P., & Giles, C. L. (2016). AlgorithmSeer: A system for extracting and searching for algorithms in scholarly big data. *IEEE Transactions on Big Data, 2*(1), 3–17. https://doi.org/10.1109/TBDATA.2016.2546302.

Ulrich, J., Carenini, G., Murray, G. & Ng, R. (2009). Regression-based summarization of email conversations.

Vanetik, N., Litvak, M., Churkin, E., & Last, M. (2020). An unsupervised constrained optimization approach to compressive summarization. *Information Sciences, 509*, 22–35. https://doi.org/10.1016/j.ins.2019.08.079.

Venter, G., & Sobieszczanski-Sobieski, J. (2003). Particle swarm optimization. *AIAA Journal, 41*(8), 1583–1589. https://doi.org/10.2514/2.2111.

Verma, P. & Om, H. (2019). Collaborative ranking-based text summarization using a metaheuristic approach. Paper presented at the Emerging Technologies in Data Mining and Information Security, Singapore.

Vijay Kumar, N. & Janga Reddy, M. (2019). Factual instance tweet summarization and opinion analysis of sport competition. Paper presented at the Soft Computing and Signal Processing, Singapore.

Vilca, G. C. V. & Cabezudo, M. A. S. (2017). A study of abstractive summarization using semantic representations and discourse level information. Paper presented at the 20th International Conference on Text, Speech, and Dialogue, Prague, Czech Republic.

Vodolazova, T. & Lloret, E. (2019). The impact of rule-based text generation on the quality of abstractive summaries. Paper presented at the Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019), Varna, Bulgaria.

Vodolazova, T., Lloret, E., Muñoz, R. & Palomar, M. (2013a). Extractive text summarization: Can we use the same techniques for any text? Paper presented at the Natural Language Processing and Information Systems, Berlin, Heidelberg.

Vodolazova, T., Lloret, E., Muñoz, R., & Palomar, M. (2013b). Extractive text summarization: Can we use the same techniques for any text? In E. Métais, F. Meziane, M. Saraee, V. Sugumaran & S. Vadera (Eds.), Natural language processing and information systems: 18th international conference on applications of natural language to information systems, NLDB 2013, Salford, UK, June 19–21, 2013. Proceedings (pp. 164–175). Berlin, Heidelberg: Springer Berlin Heidelberg.

Wali, W., Gargouri, B., & Ben Hamadou, A. (2017). Enhancing the sentence similarity measure by semantic and syntactico-semantic knowledge. *Vietnam Journal of Computer Science, 4*(1), 51–60. https://doi.org/10.1007/s40595-016-0080-2.

Wang, S., Zhao, X., Li, B., Ge, B. & Tang, D. (2017). Integrating extractive and abstractive models for long text summarization. Paper presented at the 2017 IEEE International Congress on Big Data (BigData Congress).

Wang, Y. & Ma, J. (2013). A Comprehensive method for text summarization based on latent semantic analysis. In G. Zhou, J. Li, D. Zhao & Y. Feng (Eds.), Natural language processing and Chinese computing: Second CCF conference, NLPCC 2013, Chongqing, China, November 15–19, 2013, Proceedings (pp. 394–401). Berlin, Heidelberg: Springer Berlin Heidelberg.

Warule, P. D., Sawarkar, S. D. & Gulati, A. (2019). Text summarization using adaptive neuro-fuzzy inference system, Singapore.

Woodsend, K. & Lapata, M. (2010). Automatic generation of story highlights. Paper presented at the Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden.

WordNet. https://wordnet.princeton.edu/ (Last Accessed: 26/4/2020)

WordNet: An Electronic Lexical Database (1998). (C. Fellbaum Ed.): MIT Press.

Wu, Z., Lei, L., Li, G., Huang, H., Zheng, C., Chen, E., & Xu, G. (2017). A topic modeling based approach to novel document automatic summarization. *Expert Systems with Applications, 84*(Supplement C), 12–23. https://doi.org/10.1016/j.eswa.2017.04.054.

Yadav, N. & Chatterjee, N. (2016). Text summarization using sentiment analysis for DUC data. Paper presented at the 2016 International Conference on Information Technology (ICIT).

Yao, K., Zhang, L., Luo, T., & Wu, Y. (2018). Deep reinforcement learning for extractive document summarization. *Neurocomputing, 284*, 52–62. https://doi.org/10.1016/j.neucom.2018.01.020.

Yousefi-Azar, M., & Hamey, L. (2017). Text summarization using unsupervised deep learning. *Expert Systems with Applications, 68*, 93–105. https://doi.org/10.1016/j.eswa.2016.10.017.

Yulianti, E., Chen, R., Scholer, F., Croft, W. B., & Sanderson, M. (2018). Document summarization for answering non-factoid queries. *IEEE Transactions on Knowledge and Data Engineering, 30*(1), 15–28. https://doi.org/10.1109/TKDE.2017.2754373.

Zeng, W., Luo, W., Fidler, S. & Urtasun, R. (2016). Efficient summarization with read-again and copy mechanism. Paper presented at the ICLR 2017. https://arxiv.org/pdf/1611.03382.pdf (Last Accessed: 26/4/2020)

Zhang, J., Li, K., & Yao, C. (2018). Event-based summarization for scientific literature in Chinese. *Procedia Computer Science, 129*, 88–92. https://doi.org/10.1016/j.procs.2018.03.052.

Zhong, Y., Tang, Z., Ding, X., Zhu, L., Le, Y., Li, K. & Li, K. (2017). An improved LDA multi-document summarization model based on TensorFlow. Paper presented at the 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI).

Zhou, X., Wan, X., & Xiao, J. (2016). CMiner: Opinion extraction and summarization for Chinese microblogs. *IEEE Transactions on Knowledge and Data Engineering, 28*(7), 1650–1663. https://doi.org/10.1109/TKDE.2016.2541148.

Zhu, J., Zhou, L., Li, H., Zhang, J., Zhou, Y. & Zong, C. (2017). Augmenting neural sentence summarization through extractive summarization. Paper presented at the Natural Language Processing and Chinese Computing, Dalian, China.

Wafaa S. El-Kassas is currently working toward her Ph. D. degree in Computer and Systems Engineering at Ain Shams University. She received her B.Sc. degree in Computer Engineering from Cairo University, Faculty of Engineering, Computer Engineering Department, Giza, Egypt. Eng. Wafaa received her M.Sc. degree in Computer and Systems Engineering from Ain Shams University, Faculty of Engineering, Computer and Systems Engineering Department, Cairo, Egypt in 2015. She is a journal and conference reviewer. Her research interests include text mining and analytics, automatic text summarization, natural language processing, artificial intelligence, mobile platforms, cross-platform mobile development, programming languages, source code generation, software engineering, and database systems.

Cherif R. Salama received his B.Sc. and M.Sc. degrees from the computer and systems engineering department of Ain Shams University (ASU) in 2001 and 2006 respectively. In 2010, he received his Ph.D. degree in computer science from Rice University, Houston, Texas. He worked as an assistant professor in the Computer and Systems Engineering Department of ASU and as an adjunct lecturer in the EELU, the MIU, and the GUC. He served as unit head of the Computer Engineering and Software Systems program at Ain Shams University. He is currently an assistant professor in the Computer Science and Engineering Department at the American University in Cairo. His research interests and publications span a wide spectrum including computer architecture, CAD, hardware description languages, programming languages, parallel computing, and AI.

Ahmed A. Rafea obtained his B.Sc. in Electronics and Communication Engineering from Cairo University and Ph.D. in Computer Science from University Paul Sabatier, in Toulouse, France. He worked as Professor Chair and Vice Dean of Faculty of Computers and Information at Cairo University and is currently a Professor and Chair of the Computer Science and Engineering Department at the American University in Cairo. Dr. Rafea reviewed many papers in several international journals and conferences. He was the Principal Investigator of many projects on machine translation, text mining, sentiment analysis, and knowledge engineering in collaboration with American and National Universities and Institutions. He has authored over 200 papers in conference proceedings, book chapters, and international and national journals. His research interests include natural language processing, machine translation, knowledge engineering, knowledge discovery, and data, text and web mining.

Hoda K. Mohamed is a Professor at the Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University from 2009 till now. Dr. Hoda obtained her M.Sc. from the Faculty of Engineering, Ain Shams University in 1983, her Ph.D. from the Faculty of Engineering, Ain Shams University in 1992, and was promoted to Assoc. Prof. in 2001. She is a reviewer for the IEEE International Conference on Computer Engineering and Systems, Cairo, Egypt. She has authored about 50 papers since 2009. Her research interests are on intelligent systems, E-learning systems, data mining, database systems, software engineering, natural language processing, cloud computing, and image processing.