Computing and Informatics, Vol. 43, 2024, 1185–1218, doi: 10.31577/cai_2024_5_1185

COMPREHENSIVE REVIEW OF AUTOMATIC TEXT SUMMARIZATION TECHNIQUES

Daniel O. Cajueiro

Department of Economics, Universidade de Brasília (UnB), Brazil
&
Nacional Institute of Science and Technology for Complex Systems (INCT-SC)
Universidade de Brasília (UnB), Brazil
&
Machine Learning Laboratory in Finance and Organizations (LAMFO)
Universidade de Brasília (UnB), Brasília, Brazil
e-mail: `danielcajueiro@gmail.com`

Arthur G. Nery

Department of Economics, Universidade de Brasília (UnB), Brazil
&
Machine Learning Laboratory in Finance and Organizations (LAMFO)
Universidade de Brasília (UnB), Brasília, Brazil

Igor Tavares

Mechanic Engineering Department, Universidade de Brasília (UnB), Brazil

Maísa K. De Melo

Department of Mathematics, Instituto Federal de Minas Gerais, Brazil
&
Machine Learning Laboratory in Finance and Organizations (LAMFO)
Universidade de Brasília (UnB), Brasília, Brazil

Silvia A. dos Reis

Business Department, Universidade de Brasília (UnB), Brazil

Li Weigang

Computer Science Department, Universidade de Brasília (UnB), Brazil

Victor R. R. Celestino

Business Department, Universidade de Brasília (UnB), Brazil
&
Machine Learning Laboratory in Finance and Organizations (LAMFO)
Universidade de Brasília (UnB), Brasília, Brazil

Abstract. Automatic Text Summarization (ATS) is a fundamental aspect of Natural Language Processing (NLP) that allows for the conversion of lengthy text documents into concise summaries that retain the essential information based on specific criteria. In this paper, we present a literature review on the topic of ATS, which includes an overview of the various approaches to ATS, categorized by the mechanisms they use to generate a summary. By organizing these approaches based on their underlying mechanisms, we provide a comprehensive understanding of the current state-of-the-art in ATS systems.

Keywords: Machine learning, natural language processing, summarization

Mathematics Subject Classification 2010: 68Txx, 68Uxx

1 INTRODUCTION

Automatic Text Summarization (ATS) is the automatic process of transforming an original text document into a shorter piece of text, using techniques of Natural Language Processing (NLP), that highlights the most important information within it, according to a given criterion. There is no doubt that one of the main usefulness of ATS systems is that they directly address the information overload problem [1]. They allow a possible reader to understand the content of the document without having to read it entirely.

The seminal work in ATS systems field is due to [2] that used an approach that mixes information about the frequency of words with some heuristics to summarize the text of scientific papers. There are several different approaches to designing ATS systems today. In this paper, we intend to present a comprehensive literature review on this topic. Our paper naturally relates to other reviews about this theme. We may classify these reviews in terms of classical such as [3] and [4], topic-specific such as [5] (query-based summarization) and [6] (deep learning approaches to summarization), and general reviews like ours such as [7] and [8]. Although these later works are very related to ours in terms of general content, the presentation of our work is very different. The models and mechanisms used to build such summaries drive our presentation. Thus, our focus on models and mechanisms used in automatic text summarization aims to provide practical guidance for researchers or practitioners who are developing such systems. By emphasizing these aspects of summarization, our review has the potential to offer unique insights that are not covered by other works in the field, and it may help to bridge the gap between the technique used to build the model and the practical application in summarization.

We organize the manuscript as follows: Section 2 presents a taxonomy used to classify ATS systems. In Section 3, we present the approaches to extractive summarization. We split this section into the following subsections: Subsection 3.1 presents the frequency-based methods. Subsection 3.2 presents the linguistic-based methods. Subsection 3.3 presents the methods based on supervised machine learning models. Subsection 3.4 presents the reinforcement-learning-based approaches. Section 4 presents the approaches to abstractive summarization. We divide this section into two subsections. While Subsection 4.1 introduces the linguistic approaches, Subsection 4.2 describes the deep learning sequence-to-sequence approaches. Section 5 introduces the compressive extractive hybrid approaches. Finally, Section 6 presents the main conclusions of this work.

2 CLASSIFICATION OF ATS SYSTEMS

This section presents some of the different criteria used to build a taxonomy for ATS systems [9, 10]:

1. The type of output summary: We may classify a summary into *extractive*, *abstractive* and *hybrid*. While an extractive approach extracts from the text the most important sentences and joins them in order to form the final summary, an abstractive method extracts the main information from the text and rewrites it in new sentences to form the summary. On the other hand, a hybrid approach combines ingredients of both approaches. A compressive extractive approach extracts the most relevant sentences in the first step and requires a language model in order to compress the sentences using only essential words in the second step.
2. The type of available information: We may classify a summary into *indicative* or *informative*. While the former case calls the attention of the reader to the content we may find in a text document, the objective of the latter case is to present the main findings of the text.
3. The type of content: We may classify a summary into *generic* and *query-based*. While a query-based system intends to present a summary that focuses on keywords previously fed by the user, the generic summary is the opposite.

4. The number of input documents: We may classify the summary in terms of being a *single*-document or a *multi*-document summary. The first case happens when the summary is built from only one document and the second case happens when the summary comes from the merging of many documents. A more recent approach to multi-document summarization is known as *update summarization.* The idea of update-based summarization is to generate short multi-document summaries of recent documents under the assumption that the earlier documents were previously considered.

   We may find another kind of multi-document summarization if we consider jointly to summarize the original document and the content generated by the users (such as comments or other online network contents) after the publication of the original document. This approach of summarization is known as *social context summarization.*

5. The type of method used to choose the sentences to be included in the summary: We may classify it in terms of being a *supervised* or an *unsupervised* method. This is particularly important because while in the former case we need data to train the model, in the latter case that is not necessary.

3 EXTRACTIVE SUMMARIZATION

Since the idea behind extractive summarization is to build a summary by joining important sentences of the original text, the two essential steps are

1. to find the important sentences and
2. to join the important sentences.

In this section, we show that we can use different methods to implement these tasks.

3.1 Frequency-Based Methods

In this section, we will provide a comprehensive guide to the various frequency-based approaches to summarization, assisting the reader in understanding the different techniques available.

3.1.1 Vector-Space-Based Methods

The vector space model is a model that represents each document of a collection of N_S sentences by a vector of dimension N_V, where N_V is the number of words (terms) in the vocabulary. In order to define precisely the vector space model, we start by defining the sentence-term matrix $\mathbf{M}_{\text{tfisf}}$. It is a $N_S \times N_V$ matrix that establishes a relation between a term and a sentence:

$$\mathbf{M}_{\text{tfisf}} = \begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_{N_S} \end{array} \begin{array}{c} \begin{array}{cccc} w_1 & w_2 & \cdots & w_{N_V} \end{array} \\ \begin{bmatrix} \omega_{1,1} & \omega_{1,2} & \cdots & \omega_{1,N_V} \\ \omega_{2,1} & \omega_{2,2} & \cdots & \omega_{2,N_V} \\ \vdots & \vdots & \cdots & \vdots \\ \omega_{N_S,1} & \omega_{N_S,2} & \omega_{N_S,3} & \omega_{N_S,4} \end{bmatrix} \end{array}, \tag{1}$$

where each row is a sentence and each column is a term. The weight $\omega_{j,i}$ characterizes term importance. We consider that $\omega_{j,i}$ depends on three factors. The first factor, the so-called *local factor*, relates to the term frequency. The second factor, the so-called *global factor*, relates to the sentence frequency. Finally, the third factor relates to the normalization. Thus, we may write the weight as

$$\omega_{j,i} = \frac{\widetilde{\omega}_{j,i}}{\text{norm}_j}, \tag{2}$$

where

$$\widetilde{\omega}_{j,i} = \begin{cases} f_{\text{tf}}(\text{tf}_{i,j}) \times f_{\text{isf}}(\text{sf}_i), & \text{if } \text{tf}_{i,j} > 0, \\ 0, & \text{if } \text{tf}_{i,j} = 0. \end{cases} \tag{3}$$

In Equation (2), norm_j is a sentence length normalization factor to compensate undesired effects of long sentences. In Equation (3), $f_{\text{tf}}(\text{tf}_{i,j})$ is the weight associated with the term frequency and $f_{\text{isf}}(\text{sf}_i)$ is the weight associated with the sentence frequency. We may find common choices for f_{tf}, f_{isf} and norm_j in [11]. The term frequency (TF) and inverse sentence frequency (ISF) weighing scheme, called TF-ISF, are the most popular weights in information retrieval.

Before presenting the methods, it is worth mentioning the study presented by [12] that stresses the roles of three different important dimensions that arise in frequency-based attempts to summarization:

1. Word frequencies (or some variation based on TF-ISF): They are the starting points to identify the keywords in any document (or clusters of documents);
2. The composition function: It is necessary to estimate the importance of the sentences as a function of the importance of the words that appear in the sentence.
3. The adjustment of frequency weights based on context: It is fundamental since the notion of importance is not static. A sentence with important keywords must be included in the summary as long as there is no other very related sentence with similar keywords in the summary.

Therefore, Equations (2) and (3) allow different options to select the most important terms in the complete text. In order to identify the most relevant sentences of the text, as above-mentioned, we need to aggregate these measures of importance associated with the terms and also avoid the chosen sentences having the same terms. A simple way to aggregate these measures for each term is to evaluate the average of each term in a sentence. However, in order to avoid the repetition of terms in different selected sentences, after selecting a given sentence, we may penalize the choice of sentences with terms that arise in sentences that had been previously selected.

The SumBasic method [13] uses the raw document frequency to identify the most important terms and the relevance of each sentence is given by the average of its terms. To prevent the selection of highly correlated sentences, we first select a sentence based on this criterion. Then, we square the frequencies of all terms that appear in the selected sentence, and use these updated values to reassess the relevance of the sentences. We may find an extension of [13] in [12]. In this paper, using also the frequency of the terms as an input, the authors consider different possibilities for the evaluation of the score associated with each sentence such as

1. multiplication of the frequency of the sentence's terms;
2. addition of these frequencies and the division by the number of terms in the sentence;
3. addition of these frequencies.

Note that while 1. favors short sentences, 3. favors longer sentences, and 2. is a combination of both. As in SumBasic, they also consider an additional step in the algorithm to reduce redundancy. After a sentence has been selected for inclusion, the frequencies of the terms for the words in the selected sentences are reduced to 0.0001 (a number close to zero) to discourage sentences with similar information from being chosen again.

SumBasic+ due to [14] is also a direct extension of the work of [12], in which a linear combination of the unigram and bigram frequencies is explored. They choose the parameters of the linear combination in order to maximize the ROUGE score. This work also explores query-based summarization and update-based summarization. While the setup used for update-based summarization is essentially the same, in order to attend to the task of query-based summarization, the authors suggest adding more probability mass on terms that arise in the query vector.

Using the same principle described in [13] and [12], we may use any weight as defined by Equations (2) and (3) to identify the most relevant terms and sentences and consider different methods to avoid redundancy. For instance, in order to avoid the selection of highly correlated sentences, [15] suggest that we may evaluate the relevance of each sentence by the convex combination between the relevance of the sentence given by TF-ISF terms and the maximal correlation of the sentence and the sentences already included in the summary.

An interesting way to reduce the redundancy of sentences in the final summary is to separate similar sentences into clusters. A simple way to do that is to characterize the sentences with TF-ISF vectors, use a clustering method to split the document into groups of similar sentences, and choose the most relevant sentence of each group as the one that is the closest to the centroid of each cluster [16]. These sentences are the candidate sentences to be included in the summary. We select these sentences in order of relevance based on TF-ISF.

Another interesting approach called KL Sum is to choose sentences that minimize the Kullback-Leibler divergence (relative entropy) between the frequency of words in the summary and the frequency of words in the text [17], where the sentences are greedily chosen.

A very interesting multi-document extractive approach is the submodular approach due to [18]. They formulate the problem as an optimization problem using monotone nondecreasing submodular set functions that depend on the weights introduced in Equations (2) and (3). A submodular function f on a set of sentences $\mathcal{S}$ satisfies the following property: for any $A \subset B \subset \mathcal{S} \backslash s$, we have $f(A+s) - f(A) \geq f(B+s) - f(B)$, where $s \in \mathcal{S}$. Note that f satisfies the so-called diminishing returns property and it captures the intuition that adding a sentence to a small set of sentences, like the summary, makes a greater contribution than adding a sentence to a larger set. The objective is then to find a summary that maximizes the diversity of the sentences and the coverage of the input text.

3.1.2 Matrix Factorization Based Methods

The idea of the matrix factorization methods of extractive summarization is to decompose the sentence-term matrix presented in Section 3.1.1 into a dense representation, where each term is represented by a feature (or concept). The point is that many different terms present very similar concepts. So, instead of dealing individually with the terms, we may deal directly with the concepts. In particular, in the case of ATS, we can select sentences that are good representations of different concepts that arise in the text.

Suppose that we have a text that we want to summarize with N_S sentences. We may represent this text by the sentence-term matrix $\mathbf{M}^S$ in Equation (1) with N_S rows and N_V columns. Using for instance, Singular Value Decomposition (SVD) [19, 20] decompose this matrix in the following way:

$$\mathbf{M}^S_{\text{tfisf}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{4}$$

where $\mathbf{U}$ and $\mathbf{V}^T$ are respectively orthogonal matrices of eigenvectors derived from sentence-sentence and term-term covariance matrices[1], and $\mathbf{\Sigma}$ is an $r \times r$ diagonal matrix of singular values where $r = \min(N_S, N_V)$ is the rank of $\mathbf{M}_{\text{tfisf}}$. Note that, in this representation, the rows of the matrix $\mathbf{U\Sigma}$ contain the r-dimensional representation of the N_S sentences, where each column of $\mathbf{V}$ is a base vector where each sentence is represented. Therefore, each column of $\mathbf{U\Sigma}$ is associated with a concept.

If we want that the summary finds the most important sentences of each concept, the basic idea is to select the sentences that, for each column, have the maximal absolute entry as described in [21].

[22] introduce an interesting modification to [21]'s method. It calls our attention that the latter presents two significant disadvantages: First, it is necessary to use

[1] This means that the columns of $\mathbf{U}$ are eigenvectors of $\mathbf{M}_{\text{tfisf}}\mathbf{M}^T_{\text{tfisf}}$ and the columns of $\mathbf{V}$ are eigenvectors of $\mathbf{M}^T_{\text{tfisf}}\mathbf{M}_{\text{tfisf}}$.

the same number of dimensions as the number of sentences we want to choose for a summary. However, we know that the higher the number of dimensions of the concept space, less significant topics are introduced in the summary. Second, the sentences that have a higher entry in a given concept are chosen, but they are not necessarily the most important sentences (since some of the concepts may not be that important). Therefore, the idea here is to extract the most relevant sentences s in terms of the weights $\sqrt{\sum_{k=1}^{r} u_{sk}^2 \sigma_k^2}$. In order to deal with a multi-document update summarization task, [23] create sets of topics for both previous documents and new documents. Sentences containing novel and significant topics may be extracted for building the update. Novelty is measured by the average of the internal product between the topics of the previously known documents and the topics of the new documents.

Other interesting approaches use the Non-Negative Matrix Factorization technique (NMF) [24]. The idea of NMF is to decompose $\mathbf{M}_{\text{tfisf}}^{S}$ in a product of other two matrices W and H, where we may interpret the columns of the product matrix as linear combinations of the column vectors in W using the coefficients provided by the columns of H. In general, we assume that the number of columns of W (or the number of rows of H) is lower than those of the product matrix we are decomposing. One simple algorithm to find this decomposition is based on the non-negative least squares, where we minimize the distance using the Frobenius norm between the product matrix and the actual matrices W and H. Thus, we may extract the sentences that maximize each of the topics of the documents that are the ones that for each column has the maximal absolute entry as in [21]. This is the algorithm considered in [25].

In order to deal with a query-based task, [26] provide an algorithm that follows the same steps of [25] and extracts the sentences that maximize the topics of the document that have higher similarity with the provided query.

3.1.3 Graph Based Methods

A graph (or a network) is a pair $G = (V, E)$, where V is a set where each element is called a vertex (node) and E is a set of paired vertices, where each element is called an edge. Networks are used worldwide to model systems whose components interact with each other [27]. Over the years, several metrics were introduced to characterize the networks and also the components of these systems [28]. Among them, we may cite centrality, which we use in this section, that measures the importance of each component in the network.

In the graph-based methods approach, we represent the text as a network. In this network, each sentence is a node of the network. There is a link between two nodes if the sentences are similar. Although we may think in different ways to weigh the links of these networks, simple ideas are:

1. A link exists when two sentences share a word;
2. A link exists when the similarity between two sentences exceed a given threshold.

With these definitions in mind, the most relevant sentences are those with the highest centrality and are the ones that should be included in the summary. This is the basic idea of Text Rank [29] and Lex Rank [30] that use respectively the page rank [31] and the usual eigenvector approach [32] to evaluate the centrality of the sentences in a multi-document setup.

In order to deal with a query-based approach, [33] basically uses the approach of the Lex Rank method [30]. However, instead of selecting the sentences with the highest centrality, they select sentences based on a mixture model that also considers the relevance of the sentences according to the provided query.

In [34], in a multi-document setup, the sentence similarities are built using the cosine similarity of the TF-IDF vectors of the sentences in an approach that differentiates sentences of the same document from sentences of different documents. These similarities are normalized to define a Markov chain in the network and, based on it, they evaluate the centrality of each sentence (node). In order to choose the sentences to be extracted, they penalize sentences that are very connected with the previously selected sentences.

3.1.4 Topic-Based Methods

The topic-based summarization methods rely on topic representations such as the Latent Dirichlet Allocation (LDA) [35]. LDA is a generative model[2] that represents a document by a collection of topics and each topic, in its turn, by a collection of words.

In order to develop the so-called TopicSum, [17] assume a fixed vocabulary V and propose a LDA-like generative model. For the sake of organization, although this approach was originally developed for multi-document summarization, we start here with a setup for single-document summarization:

1. Draw a "background" vocabulary distribution ϕ_B from Dirichlet(V, λ_B) shared across the document collection representing the background distribution over vocabulary words.
2. For each document d, we draw a "content" distribution ϕ_C from Dirichlet(V, λ_C) representing the significant content of d that we wish to summarize.
3. For each sentence s of each document d, draw a distribution ψ_T over topics (content, background) from a Dirichlet prior with pseudo-counts (n_C, n_B), where $n_C < n_B$ reflects the intuition that most of the words in a document come from the background.

Using this generative model, the authors [17] estimate ϕ_C for each document and select the most important sentences using the same criterion used by the KLSum discussed in Section 3.1, replacing the frequency of the words in the text, which is a unigram distribution, by ϕ_C. An extension of this model, called HIERSum and

[2] A generative model is a model that describes the distribution of the data and tells how likely a given example is.

provided by the same work, considers that a document may be formed by different topics as in [36].

3.1.5 Neural Word Embedding Based Methods

A word embedding is a vector that represents a word [37]. In general, there are several features used to capture the essence of the word. Although we may also consider the vectors, that aggregate the words by concepts, generated in Section 3.1.2 kinds of word embeddings representation, the literature usually refers to word embeddings as the representations based on neural networks models that extend classical models of language.

Several papers propose different ways to find word representations (or word embeddings). The most popular algorithms that can generate word embeddings from text are Word2vec [38], GloVe [39], and fastText [40]. All of these models are neural probabilistic language models. We train them in a supervised fashion. This means that we build inputs and outputs for the neural networks using all the sentences and all the words in these sentences, that are available in the training documents without the need for annotated texts. However, they differ from each other on the performance index, the type of input, the type of output, and how they weigh local and global information about each word that arises in the documents. While some of these models use performance indexes that are variations of the likelihood functions associated with multinomial logit models (a kind of cross-correlation entropy), others use variations of the mean square error. In order to understand the different types of inputs and outputs, consider, for instance, a sentence formed by a sequence of words "You get a shiver in the dark." that is the first sentence of the song Sultans of Swing by the British band Dire Straits. Using sequences of words of this kind, we may train, for instance, a Continuous Bag Of Words (CBOW) model. Using this model, we want to predict the word "*shiver*" using its neighborhood (context) of size 2 formed by the words "get", "a", "in" and "the". Another possibility is to use the skip-gram model and to use the word "*shiver*" to predict its neighborhood given by "get", "a", "in" and "the". Another interesting approach is due to [41] (CW) who train a neural network to differentiate between a valid n-gram and a corrupted one. Furthermore, in these examples, we may note that we are only using local information. However, as we mentioned before, we may also use global information given by, for instance, the term-document matrix (similar to the term-sentence matrix considered in Section 3.1.1) to weigh the performance index. All these algorithms have some design and hyperparameter choices and we may find very different results depending on them [42, 43]. In these models, the input and output layers have their sizes given by the vocabulary that arises in the collection of documents. The so-called neural word embedding associated with a word is the average of the vectors of parameters associated with the input and the output of this word.

There are several methods that we can use to consider the information encapsulated in word embeddings. The main motivation behind the use of word embeddings is to deal with the main drawbacks of the space vector models approach associated with the fact that similar words are treated separately:

1. Similar words may have very different rankings. Therefore, it fails to assign appropriate scores to the sentences;
2. The summary may be redundant, since the sentences of the summary may come from different words that have similar use and meaning.

One of the first ideas of using word embeddings in extractive summarization is due to [44]. They use the setup of greedy submodular optimization due to [18], reviewed in Section 3.1.1, and different word embeddings (Word2Vec and CW) to extract sentences.

One of the simplest ideas is to use a kind of centroid method such as in [45]. We may review this method using the following steps:

1. Create a representation of the documents in the dataset using the vector space models;
2. For each document, identify the most relevant words, i.e., the words that have a weight (provided by the space vector model) larger than a given threshold;
3. Evaluate the centroid of each document averaging the word embeddings of each word selected in the last step;
4. Evaluate the word embedding of each sentence in a document averaging the word embeddings of each word that arises in the sentence;
5. Identify the most relevant sentences that are the sentences that are the most similar to the centroid of the document.

[46] use word embeddings to find the m most similar words to each word in a given sentence. Then each sentence is represented by a large vector of words, where each word in the original sentence is replaced by these m most similar words previously found using the word embedding representation. With this new representation of each sentence, it applies TF-ISF to this new representation of the sentences and any algorithm presented in Section 3.1.1 may be used to extract the most important sentences. In particular, they use a clustering method similar to [16].

The idea behind the work of [47] is to build a list of important words that they call keywords (first sentence words and high-frequency words) and to rank the sentences in the document according to the cosine similarity between the embeddings of the keywords and the embeddings of the words that form the sentences.

3.2 Linguistic-Based Methods

There are different attempts to extract sentences to form abstracts based on linguistic methods. We may use the same classification that [48] suggested to classify full abstractive methods to be mentioned in Section 4.1, namely *structured-based* and *semantic based*. The *structured-based* approach uses cognitive schema such as a set of rules, a template, a tree parser and a domain ontology. On the other hand, the *semantic-based* approach starts with a semantic representation of the document and uses this representation to identify the most important sentences of the document.

[49] is one of the first works to include linguistic information to decide whether a sentence should belong to the summary. The authors consider as important features the location of a sentence in the document, the location of words, or the punctuation in a sentence. They suggest that while the location of the sentence in a document is very subjective and it depends on the authors' choices, it is possible to get some pieces of relevant information using the location of words or punctuation in a sentence. Additionally, they claim that question marks should never be included in the summary for the following reasons:

1. they never provide a complete description of the facts;
2. if a question is selected, then the context behind the question should also be selected.

It is worth mentioning that this discussion about whether a piece of text should be included provides the basic ideas for abstractive summarization discussed in Section 4. On the other hand, it defends that existence of some phrases provides a powerful approach to sentence selection or rejection. For instance, words such as "Our work", "This paper", and "Present research", which are used to state the purpose of a paper, serve to indicate that such sentences should be selected for the abstract. On the other hand, opinions, or references to figures or tables such as "obvious", "believe", "Fig.", "Figure 1", and "Table IV" should not be included in an abstract.

We may find an interesting example of a structured-ontology-based approach in [50]. A domain ontology is a set of concepts and categories in a given subject area that shows their properties and the relations between them. [50] encode an ontology with a tree structure, and each node includes the concepts represented by the node's children. When the count of any node increases, the counts associated with their ancestors also increase. They use this principle to score paragraphs. After marking the counts of the nodes in the ontology, the authors select second-level nodes that have higher counts as the main subtopics of the article. In order to implement the method, the authors only consider the top n subtopics. Their system uses the obtained subtopics to select paragraphs to form the summary. They rank the paragraphs based on their "closeness" to the selected subtopics by counting the words in common between the paragraph and each selected subtopic. Since we select n possible subtopics, there are also n scores associated with each paragraph, and these n scores represent the relevance of the n paragraphs for each selected topic. They assume that the score of each paragraph is the sum of its weighed relevance to the topics.

We may find a kind of informative semantic-graph-based approach for multi-document summarization in [51]. The authors use a *semantic role labeling* parser to extract each argument of the sentence. Semantic role labeling assigns labels to words or phrases indicating their semantic role. It is often described as a technique to answer "Who did what to whom". Thus, the authors calculate the composite similarity between all semantic frames based on the event-indexing model [52] to keep track of five indices, namely temporality, spatiality, causality, and intention. Then, they generate a semantic graph where nodes are semantic frames and edges are the composite similarity values. In order to choose the most important sentences, they modify the Page Rank, used in Section 3.1.3, in order to identify the most significant edges in the graph.

[53] present an interesting semantic-lexical-chain approach that uses semantically related concepts to identify the sentences useful for extraction. A lexical chain is a sequence of semantic-related ordered words [54]. WordNet[3] is an excellent source to extract lexical chains. For example, this lexical chain was extracted from WordNet: "device $\rightarrow$ musical instrument $\rightarrow$ string instrument $\rightarrow$ guitar $\rightarrow$ electrical guitar". [53] use a "part-of-speech" tagger and a comprehensive thesaurus to map words in concepts. The sentences to be extracted are the ones that present the most important concepts.

3.3 Supervised Machine Learning-Based Methods

The objective of using supervised machine learning methods in ATS is to explore the problem of ATS as a binary classification problem in which we use supervised methods of machine learning to select the sentences that should arise in the summary using their features.

This classical and conventional approach follows the steps:

1. Tag the sentences manually in the documents as positive ones (the ones that should be extracted to build the summary) and the negative ones (the opposite).
2. Select the features associated with each sentence that are used as inputs in the classification algorithm. These features may be word-based features and sentence-based features. The word-based features may be, for instance, weights that come from the space vector representation or the word embeddings. On the other hand, the sentence-based features may be the sentence position or the sentence length.
3. Estimate the model. These models usually have hyperparameters that have to be set before the estimation of the parameters. Since there is no way to choose these hyperparameters without data experimentation, a cross-validation[4] [56] procedure is necessary.

[3] WordNet is a large lexical database of English. It groups nouns, verbs, adjectives, and adverbs into sets of cognitive synonyms, the so-called synsets, each expressing a distinct concept [55].

[4] Cross-validation is a resampling method that splits the data into different sets in order to test and train a model on different iterations.

4. Apply the classification algorithm to find out the sentences that should be included in the summary.

There are several supervised machine learning classifiers in the literature and, in essence, we can use any of them. A machine learning classifier is a mathematical model that, given a set of attributes, provides a label associated with a class. We may find a review of the supervised machine learning methods in [57].

On the other hand, modern approaches based on neural networks are able to replace the second and third steps above by the automatic selection of a "composition" of features that are able to solve the binary classification problem of extractive summarization. There are different neural network models that we can use such as multilayer perceptron, convolution neural networks and sequence-to-sequence neural network models as reviewed in Section 4.2.

The first paper that used a supervised approach to extractive summarization is [58]. This work uses several sentence features such as whether the sentences include the most frequent words of the document, whether the sentences include words presented in the title or in the list of keywords associated with the document, the position of the sentences in the document (important sentences usually arise at the beginning or the end of the document) and if the sentences contain indicator phrases such as "This report (...)".

We may find an interesting contribution to summarization in [59]. With the motivation to model the local dependencies between sentences, the authors use a Hidden Markov Model (HMM) that models the transitions between sentences that should or should not belong to the summary. They use only three features, namely the position of the sentence in the document, the number of terms in the sentence, and how likely sentence terms are in the document.

[60] calls attention to the fact that the assumption of independence of features of the Naive Bayes is a strong assumption and the maximum entropy classifier outperforms this model. In particular, this work uses the following features: *word pairs*, which simply tells whether a particular word pair is present, *sentence position*, which indicates whether the sentence belongs to the beginning, the middle or the end of the document and *discourse features*.

An interesting contribution comes from [61] that uses a feedforward neural network to find the most important sentences of a piece of text. The neural network is trained using the RankNet algorithm [62][5]. For each sentence, the work considers several features such as the position of the sentence, n-grams of words in the sentence, terms common with the title and the presence of some specific words such as in [49].

[63], besides using several of these previously discussed attributes to characterize the sentences of a document, they also use the mean of word embeddings associated with each word of a sentence as an additional attribute.

[5] RankNet is a pair-based gradient descent algorithm used to rank a set of inputs, in this case, the set of sentences in a given document.

In order to deal with a query-based task, AttSum [64] builds a convolutional neural network composed essentially of three major layers:

1. A convolutional neural network layer to project the sentences and queries onto the embeddings;
2. A pooling layer to combine the sentence embeddings to form the document embedding in the same latent space and to indicate the query relevance of a sentence;
3. A ranking layer that ranks sentences according to the similarity between its embedding and the embedding of the document cluster.

The sequence-to-sequence neural networks models [65, 66, 67], discussed in Section 4.2, different from the classical and conventional approaches, are able to automatically extract features of the sentences and represent them mathematically by internal weights of the neural network that can be used to classify the sentences as the ones that should belong to the summary.

3.4 Reinforcement Learning Based Methods

A reinforcement learning problem explores a situation where the objective is to map states to actions in order to maximize a numerical reward signal. We usually define a reinforcement learning solution with the following ingredients [68]:

1. An agent (the learner or the decision-maker) that interacts with the environment in a sequence of instants $t = 0, 1, 2, \ldots$.
2. At each instant t, the agent faces a state $s_t \in \mathcal{S}$, where $\mathcal{S}$ is a finite set of possible states, and selects an action $a_t \in \mathcal{A}(s_t)$, where $\mathcal{A}(s_t)$ is the set of admissible actions contingent to the state s_t.
3. At each state the agent receives a reward that is contingent to the chosen action $r : \mathcal{S}_t \times \mathcal{A}(s_t) \to \Re$. We usually assume that $\max_{s \in S} \max_{a \in \mathcal{A}(s)} r(s, a) < \infty$.
4. In each state, the agent maps states to probabilities of selecting a possible action. This map is called agent policy and it is denoted by Π_t, where $\Pi_t(s, a)$ is the probability that $a_t = a$ when $s_t = s$.
5. In each state s_t, the agent intends to maximize the expected value of the return given by

$$v_\gamma^\pi(s) = E^\pi \left[R_t \big/ s_t = s \right] = E^\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \Big/ s_t = s \right], \quad (5)$$

where the return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$ and $E^\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \big/ s_t = s \right]$ is the expected value of the discounted sequence of rewards received by the agent assuming that in the beginning of the trajectory it was in state s and, from this state, it followed the policy π. A common assumption is that the rewards are bounded and the distributions of rewards and states are stationary. The discount factor $0 < \gamma < 1$ has two functions. From the economic point of view, it makes explicit the value of money over time. On the other hand, from the mathematical point of view, it ensures that under mild regularity conditions, there is an optimal policy that maximizes the performance index presented in Equation (5). Another important assumption here is that we may describe the transitions between the states as a Markovian process, i.e.,

$$P(s_{t+1} = s', r_{t+1} = r/s_t, a_t, r_t, \ldots, s_0, a_0, r_0) = P(s_{t+1} = s', r_{t+1} = r/s_t, a_t, r_t). \tag{6}$$

This assumption allows a parsimonious representation that reduces the computational complexity of the problem and simplifies the definition of the transition matrices, the reward structure and the set of admissible actions.

6. If the rewards and the transition probabilities are known and stationary, we may show that the solution of the problem is given by the Bellman equation:

$$v^{\pi}_{\gamma} = \mathtt{R}_{\pi} + \gamma \mathtt{P}_{\pi} v^{\pi}_{\gamma}, \tag{7}$$

where $\mathtt{P}_{\pi} = [\mathtt{P}_{ss'}]$ is the transition matrix contingent to the policy π, $\mathtt{R}_{\pi} = [\mathtt{R}_{s}]$ is the returns vector contingent to the policy π, $\mathtt{P}_{ss'} = \sum_{a \in \mathcal{A}(s)} q^a_s P^a_{ss'}$, $\mathtt{R}^a_{ss'} = E[r_{t+1}/s_t = s, a_t = a, s_{t+1} = s']$ and q^a_s is the probability of using the action a in state s. In order to solve this problem, we need to know the value function given the policy π, which is a problem known as *policy evaluation*, and to improve the policy given another policy, which is a problem known as *policy improvement.* Note that the problem of policy evaluation is equivalent to finding the solution of a linear system, i.e., given the policy and assuming that the rewards and probabilities are stationary and known, Equation (7) is a linear system. We can prove that the operator $L(v) = \mathtt{R}_{\pi} + \gamma \mathtt{P}_{\pi} v$ is a contraction and, according to the Banach fixed point theorem, it can be solved by fixed point iterations[6]. In order to improve the policy, we use the Q-function $Q^{\pi}(s, a) = E_{\pi}[R_t/s_t, a_t = a]$, that is the expected return for taking action a in state s and thereafter following an optimal policy. Thus, if $Q^{\pi}(s, a) > v^{\pi}(s)$, then π can be improved if $\pi(s)$ is replaced by a. There are two popular algorithms to solve this problem, namely policy iteration and improvement and value iteration. While in the former approach the policy improvement and policy evaluation tasks are run in separate loops, in the latter, these are carried out in the same loop.

7. When the transition probabilities and rewards are not known, we use Monte Carlo methods to sample sequences of states, actions, and rewards. The policy evaluation and policy improvement steps are completed using average returns of episodic tasks.

8. Temporal difference learning methods combine Monte Carlo and dynamic programming approaches. They learn from experience like the Monte Carlo meth-

[6] See, for instance, [69].

ods and they update estimates like the dynamical programming approach. Two algorithms can be used to implement the temporal difference approach, namely SARSA (an Acronym for State-Action-Reward-[next]State-[next]Action) and Q-learning. While the Q-learning approach updates the value function using a greedy action approach, SARSA updates with the actual action used for generating experience by the agent.

[70], in order to formulate the problem of extractive summarization as a reinforcement learning problem, reduce the document to be summarized in a set of n sentences, define a score function for any subset of sentences of the document $S \subset d$, where S is one of the possible summaries and d is the document, and also define the length function that indicates the size of the summary. They use a temporal difference learning approach to find the summary that maximizes the score function that considers a trade-off between relevance and redundancy subject to the maximal summary length. Thus, in this problem, a state is a summary of a given length. In this work, the actions are deterministic and are basically the decision to include or not a given sentence. The reward is defined in such a way that the agent only receives the rewards when the summary reaches the final size. The score function specifically depends on the coverage of important words (the count of top-100 words in terms of the TF-IDF included), coverage ratio (the count of top-100 elements included), redundancy ratio (the counting of the number of elements that excessively cover the top 100 elements), length ratio (the ratio between the length of the summary and length limitation) and the sum of the inverse position of the sentences.

We may find an extension of the work of [70] in [71]. In a multi-document setup and also working with a query-based task, they use the SARSA algorithm, explore different types of rewards (not only delayed rewards as in the case of [70]) and also use the ROUGE to evaluate the similarity between the reference summary and the automatic summary.

4 ABSTRACTIVE SUMMARIZATION

Different from extractive summarization, which builds a summary from the combination of the important sentences previously extracted from the original text, in abstractive summarization, we need a language model to rewrite the summary from scratch.

4.1 Linguistic Approaches

As we have mentioned in Section 3.2, we may split the linguistic approaches to abstractive summarization into two large groups, *structured-based* and *semantic-based* [48].

An example of the *structured-based* approach that uses a *rule-based* scheme is [72]. In this work, in order to provide summaries in the fields of "Accidents and Natural Disasters", "Attacks, Health and Safety", "Endangered Resources", and

"Investigations/Trials", the authors use handcrafted information extraction rules, content selection heuristics and generation patterns. In particular, to extract the information they need to build the summary, they ask the following questions:

1. What: what happened;
2. When: date, time, other temporal placement markers;
3. Where: physical location;
4. Perpetrators: individuals or groups responsible for the attack;
5. Why: reasons for the attack;
6. Who was affected: casualties (death, injury), or individuals otherwise negatively affected;
7. Damages: damages caused by the attack;
8. Countermeasures: countermeasures, rescue efforts, prevention efforts, other reactions.

With the answers to these questions in hand, extracted using predefined extraction rules, they use previously generated patterns to build the summaries.

An example of the structured approach that uses a *template* for multi-document summarization is [73]. In this work, the authors use templates that represent each topic of the piece of text that need to be summarized and are populated using information extraction rules. In order to generate the summaries, they use a parse tree to identify the Subject-Verb-Object structures[7] and WordNet to classify the topic of each structure. The topics with a high frequency are included in the summary.

We may find an example of the *structured-approach* that uses a *domain ontology* in [74]. In this work, the authors extend a domain ontology using concepts of fuzzy logic by embedding a set of membership degrees in each concept of the domain ontology. They use this fuzzy domain ontology to extract the sentences related to the text domain and to generate the abstract by concatenating concepts with relations.

We may find one of the first examples of the *semantic-based* approach in [75]. This work proposes the concept of *Information Items* (INITs) to help to define the abstract representation, which is the smallest element of coherent information in a text or a sentence. The goal is to identify all entities in the text, their properties, the predicates between them, and the characteristics of the predicates. In that work, the implementation of INITs is constrained to dated and located subject-verb-object (SVO) triples and the summary generation as above-mentioned is carried out using parse trees. This work is another example of an ATS that generates informative summaries.

Another interesting example of the *semantic* based approach is the *multimodal model* of [76]. We may summarize the implementation of this model in three steps:

[7] In linguistic, subject-verb-object (SVO) is a sentence structure where the subject comes first, the verb second, and the object third.

1. Building the semantic model: The semantic model, which should consider both images and pieces of text, is built based on a knowledge representation based on a domain ontology;
2. Rating the informational content: In order to rate the content, the authors propose the so-called information density metric (ID) which rates a concept's importance based on factors such as completeness of attributes, the number of connections with other concepts and the number of expressions that put the concept in evidence;
3. Generating a summary: The work uses the concepts of TAG (Tree Adjoining Grammars) Derivation Trees[8] as in [78] to express the concepts and relationships found in previous steps.

[79] implement the *semantic-based* approach using a *semantic graph*. The method consists of three steps:

1. The creation of the semantic graph called Rich Semantic Graph (RSG) for the original document;
2. The reduction of the generated semantic graph;
3. The generation of the final abstractive summary from the reduced semantic graph.

In RSG, the verbs and nouns of the input document are represented as graph nodes along with edges corresponding to semantic and topological relations between them. The graph nodes are instances of the corresponding verb and noun classes in the domain ontology. The Rich Semantic Graph Reduction Phase aims to reduce the generated rich semantic graph of the source document to a reduced graph. A model of heuristic rules is applied to reduce the graph by replacing, deleting, or consolidating the graph nodes using the WordNet relations. Finally, the Summarized Text Generation Phase aims to generate the abstractive summary from the reduced rich semantic graph. To achieve its task, this phase accesses a domain ontology, which contains the information needed in the same domain of RSG to generate the final texts.

4.2 Sequence-to-Sequence Deep Learning Methods

Sequence-to-sequence deep neural network models have been widely used in text summarization tasks, where they take in a source text as input and generate a corresponding summary as output. To train these models, a large dataset of source texts and their corresponding summaries is required. The model is then trained on this dataset to learn the relationship between the input source text and the output summary. During training, the model tries to minimize the difference between the

[8] A TAG is a formalism that builds grammatical representations through the composition of smaller pieces of syntactic structure [77].

generated summary and the reference summary in the dataset, using a loss function such as cross-entropy or mean squared error.

In order to explore the most recent approaches of sequence-to-sequence models we need to trace back to the first architectures of Recurrent Neural Networks (RNN) [80]. RNNs are sequence models that deal directly with two modeling constraints of the standard multilayer perceptrons that are essential to model sequences. First, it is assumed that the input sequences have the same length. Second, it is an architecture that does not allow for the same token in different positions of the text to have similar features. We may summarize the vanilla RNN by the set of equations:

$$s^i = g\left(W_{ss}s^{i-1} + W_{sx}x^i + b_s\right) \tag{8}$$

and

$$y^i = g\left(W_{ys}s^i + b_y\right), \tag{9}$$

where x^i is a token in a piece of text, y^i is the token we want to predict, s^i is the state of the RNN, W_{ss}, W_{ax}, W_{ys}, b_s and b_y are the parameters of the network that we need to learn and s^0 is a vector of zeros. Like the other models of language, we try to predict the probability of the next word. Therefore, we may estimate the parameters of this model in a supervised fashion using the *backpropagation through time* algorithm. Suppose, for instance, that your text includes the first sentence of the song Africa by the band Toto "I hear the drums echoing tonight." Let x^1 = "I", x^2 = "hear" x^3 = "the", x^4 = "drums", x^5 = "echoing", y^1 = "hear" y^2 = "the", y^3 = "drums", y^4 = "echoing" and y^5 = "tonight". Thus, this algorithm tries to maximize the probability that the token "hear" arises when the token "I" is an input, to maximize the probability the token "the" happens when "hear" is the input and s_1 is the state of the system generated by "I" and so on. Unfortunately, vanilla RNNs are not good at dealing with long sequences. Problems that arise in this context are the so-called *exploding* and *vanishing* gradients [81]. This happens naturally due to the algorithm of backpropagation, which is based on the chain rule of calculus, and the consequent multiplication of the same shared matrix of the parameters several times.

In order to overcome the difficulties of exploding and vanishing gradients, two important models of RNNs were introduced, namely Long Short-Term Memory (LSTM) [82] and Gated Recurrent Units (GRU) [83]. The basic idea behind these models is to replace the simple units of the vanilla RNN model with complex units which include gates that control the flow of information that passes from one unit to the other.

A natural extension of the vanilla RNNs is to consider the case where the input sequence and output sequence have different sizes. These models are called *Encoder-Decoder* models and they aim at mapping one sequence to another sequence [84], where the encoder (decoder) is the part of the model that deals with the input (output) sequence. In this type of model, the intention is, given a sequence $\{x^i\}$, to predict another sequence $\{y^i\}$ not necessarily of the same size. For example, consider that we want to use a sequence-to-sequence model to translate the first sentence of the song “Smoke on the Water” by the English band Deep Purple to Spanish. We may have x^1 = “We”, x^2 = “all”, x^3 = “came”, x^4 = “out”, x^5 = “to”, x^6 = “Montreux” and we want to predict y^1 = “Todos”, y^2 = “salimos”’, y^3 = “a” and y^4 = “Montreux”. In text summarization, the input text is first encoded into a fixed-length vector representation by the encoder network. This vector is then passed through the decoder network, which generates the corresponding summary one term at a time.

A fundamental contribution to improving the learning process of sequence-to-sequence models is the attention mechanism [85]. The main idea behind this paper is to include a set of weights to inform the model about the context. For instance, consider again the sentence “We all came out to Montreux.” We know that the phrasal verb “come out” has different meanings. However, in this sentence, it is obvious that the sense of this verb is “to go somewhere” and this happens because of the word “Montreux”, which is a town in Switzerland.

The most recent models of NLP are based on transformers. Transformers are networks models that comprise the idea of processing complex information in parallel and an extension of the above-mentioned attention mechanism called multi-head attention [86]. Multi-head attention is the idea of evaluating several attention models simultaneously. In a given sentence, we may use several attention mechanisms to explore several different dimensions at once. For instance, the chorus of the song “America” by Neil Diamond says “They’re coming to America today.” We may think of a multi-head mechanism as a way to help us ask and answer questions, i.e., it says where we should pay attention in a sentence to answer a given question[9]. Thus, the first question could be: “What is happening?” Then, the answer is “They are coming somewhere.” The second question could be “Where?” Then, the answer is “America”. The third question could be “When?” Then, the answer is “Today”. In the transformer network, this information is encoded in vectors called *Queries* (Q), *Keys* (K), and *Values* (V). Like in the recurrent sequence-to-sequence models, the architecture of Transformer models has an encoder-decoder structure. The encoder model receives the embeddings (discussed in Section 3.1.5) of the inputs with a position encoding. The position encoding serves to inform the position of the token in the sequence, which is necessary here since this is a parallel model where the entire text is simultaneously inputted. The encoder block is formed by a stack of multi-head attention blocks connected with a feedforward network to generate the vectors Q, K and V that feed the following encoder blocks. The decoder block is a stack of an additional multi-head attention block and a block similar to the encoder block. The first block receives the output embeddings and generates the vector Q to be used together with the vectors K and V it receives from the encoder block. This model is used to generate the output sequence in a recursive fashion. There is now a large list of transformers that have been used in successful NLP tasks [87, 88].

[9] It has the same role as the filters built automatically by NN models to identify dimensions of images in CNN.

5 COMPRESSIVE EXTRACTIVE APPROACHES

Compressive extractive approaches usually depend on two steps:

1. We extract the sentences that should belong to the summary;
2. We compress the chosen sentences that come from the original text in order to present only essential information.

When selecting the sentences to be extracted, we usually rely on one of the methods already discussed in Section 3. On the other hand, in order to compress the sentences we need a model of language.

One advantage of the compressive extractive approaches over the pure extractive approaches is that for the case of the summaries that need to have a previously given constant length, the summaries created with the compressive approach usually contain more information than the summaries created with extractive approaches. This happens because by removing insignificant sentence components, we make room for more relevant information in the summary.

We have discussed many methods to extract sentences from the whole text in Section 3. On the other hand, sentence compression is by itself a research problem. It starts with the input which is a sentence with n words. The algorithm for sentence compression may drop any subset of these words and the words that remain with an unchanged order form a compression. Thus, the problem is not trivial since there are 2^n possible pieces of text to be chosen [89]. In order to solve this problem, we have to develop a method to determine what is the relevant piece of information in a sentence and how to present this information grammatically.

In order to implement the compressive step of compressive extractive summarization, we may use unsupervised, supervised methods or hybrid approaches.

The unsupervised approaches delete words based on part-of-speech tags[10] or the lexical items[11] alone. In particular, we may find a very interesting approach for unsupervised compressive summarization in [90]. In order to generate a summary, their approach focuses on extracting topic words, weighing correct-word concatenations linguistically, and extracting reliable components of speech recognition acoustically as well as linguistically. A set of words maximizing a summarization score, indicating the appropriateness of a summarized sentence, is selected from those using a Dynamic Programming (DP) technique. The summarization score consists of word significance measured by the frequency of each word in the sentence, word confidence measured by the logarithm of the probability of n-grams, and the linguistic likelihood of summarized sentences.

[10] In linguistics, Part-Of-Speech (POS) tags, are tags used in a text (corpus) to associate a given word with its corresponding part of speech, based on both its definition and its context.

[11] A lexical item may be a single word, a part of a word, or a sequence of words that forms the basic elements of a language's vocabulary.

Another interesting approach is the graph-based approach due to [91] where a directed word graph is constructed. In this digraph, nodes represent words and edges represent the adjacency between words in a sentence. Thus, the authors can compress sentences by finding the k-shortest paths in the digraph.

The supervised approaches may depend on a number of resources such as an annotated corpus with the original sentences and their corresponding reduced forms written by humans for training and testing purposes, a lexicon[12] and a syntactic parser to generate a parse tree[13].

[92] focuses specifically on the problem of sentence reduction of extracted sentences for summarization. The author assumes that the input of his system is the collection of extracted sentences that we can build using one of the methods of Section 3. On the other hand, his algorithm of sentence reduction has five steps:

1. Syntactic parsing: He parses the input sentence to produce the sentence parse tree;
2. Grammar checking: He determines which components of the sentence must not be deleted to keep the sentence grammatical. To do this, he traverses the parse tree generated in the first step in top-down order and marks, for each node in the parse tree, which of its children are grammatically obligatory;
3. Context information: The system decides which components in the sentence are most related to the main topic being discussed. To measure the importance of a phrase in the local context, the system relies on lexical links between words;
4. Corpus evidence: The program uses the annotated corpus consisting of sentences reduced by human professionals and their corresponding original sentences to compute how likely (measuring the probabilities) are humans to remove a certain phrase;
5. Final decision: The final reduction decisions are based on the results from all the earlier steps.

To decide which phrases to remove, the system traverses the annotated sentence parse tree and removes a phrase when it is not grammatically obligatory, not the focus of the local context and has a reasonable probability of being removed by humans.

Another interesting approach to supervised compressive summarization explores the noisy channel framework [93]. In this framework, the authors consider that every

[12] A lexicon is the vocabulary of a language or a subject. For instance, the lexicon of computer science must present keys such as “algorithm”, “big data”, “class”, “design pattern” and so on.

[13] A syntactic parsing converts the sentence into a tree whose leaves hold POS tags, but the rest of the tree tells how exactly these words join together to make the complete sentence. For example, a linking verb and a verb may combine to be a Verb Phrase (VP) such as in “I $\underline{\text{have been studying}}$ English for years.”, where the underlined piece of text is a verb phrase.

sentence was originally shorter and then someone added some additional noisy text to it. Thus, the task of compressive summarization is to find the original sentence. It is worth mentioning that it is not relevant here whether or not the "original" string is real or hypothetical. In this approach, the authors split the problem of sentence compression into three sub-problems:

1. Source model: They assign to every string (sentence) s a probability $P(s)$, which gives the chance that s is generated as an "original short string";
2. Channel model: They assign to every pair of strings (sentences) s and t a probability $P(t|s)$, which gives the chance that the expansion of the short string s results in the long string t;
3. Decoder: They search for the short string s that maximizes $P(s|t) = P(s)P(t|s)$.

In order to implement this, they assume that the probabilities $P(s)$ and $P(t|s)$ are associated with the representation of these sentences using parse trees.

In [94] and [95] different from the above-mentioned works deal with both the extractive and compressive steps using neural network models such as the ones presented in Section 4.2.

Although the most common compressive approaches focus on editing sentences using compressive operations, other operations are also possible. [96], based on analysis of human written abstracts, call attention to different types of operations such as *sentence combination* (merging material from several sentences), *syntactic transformation* (for instance, to change the position of the subject or to transform a piece of text from passive voice to active voice), *lexical paraphrasing* (replacing phrases with their paraphrases), *generalization or specification* (replacing phrases or clauses with more general or specific descriptions) and *reordering* (changing the order of specific sentences).

6 FINAL REMARKS

In this work, we have provided a literature review of ATS systems. As we have previously mentioned, this is not an easy task. Since work by [2], thousands of papers were introduced about this subject. As we mention before, we drive our presentation using the models and mechanisms used to build the summaries. Several models have been used to generate summaries including the *classical frequency-based models* and the state of art *deep neural network sequence-to-sequence models.*

Acknowledgment

The authors acknowledge and thank the partial financial support from the Ministry of Science and Technology of Brazil (MCTI). DOC (grant No. 302629/2019-0) and LW (No. 309545/2021-8) also thank CNPQ for the partial financial support.

REFERENCES

[1] Edmunds, A.—Morris, A.: The Problem of Information Overload in Business Organisations: A Review of the Literature. International Journal of Information Management, Vol. 20, 2000, No. 1, pp. 17–28, doi: 10.1016/S0268-4012(99)00051-1.

[2] Luhn, H. P.: The Automatic Creation of Literature Abstracts. IBM Journal of Research and Development, Vol. 2, 1958, No. 2, pp. 159–165, doi: 10.1147/rd.22.0159.

[3] Edmundson, H. P.—Wyllys, R. E.: Automatic Abstracting and Indexing – Survey and Recommendations. Communications of the ACM, Vol. 4, 1961, No. 5, pp. 226–234, doi: 10.1145/366532.366545.

[4] Paice, C. D.: Constructing Literature Abstracts by Computer: Techniques and Prospects. Information Processing & Management, Vol. 26, 1990, No. 1, pp. 171–186, doi: 10.1016/0306-4573(90)90014-S.

[5] Rahman, N.—Borah, B.: A Survey on Existing Extractive Techniques for Query-Based Text Summarization. 2015 International Symposium on Advanced Computing and Communication (ISACC), IEEE, 2015, pp. 98–102, doi: 10.1109/ISACC.2015.7377323.

[6] Alomari, A.—Idris, N.—Sabri, A. Q. M.—Alsmadi, I.: Deep Reinforcement and Transfer Learning for Abstractive Text Summarization: A Review. Computer Speech & Language, Vol. 71, 2022, Art. No. 101276, doi: 10.1016/j.csl.2021.101276.

[7] Mridha, M. F.—Lima, A. A.—Nur, K.—Das, S. C.—Hasan, M.—Kabir, M. M.: A Survey of Automatic Text Summarization: Progress, Process and Challenges. IEEE Access, Vol. 9, 2021, pp. 156043–156070, doi: 10.1109/ACCESS.2021.3129786.

[8] El-Kassas, W. S.—Salama, C. R.—Rafea, A. A.—Mohamed, H. K.: Automatic Text Summarization: A Ccomprehensive Survey. Expert Systems with Applications, Vol. 165, 2021, Art. No. 113679, doi: 10.1016/j.eswa.2020.113679.

[9] Sparck Jones, K.: Automatic Summarising: Factors and Directions. In: Mani, I., Maybury, M. (Eds.): Advances in Automatic Text Summarization. MIT Press, 1998, pp. 1–12, doi: 10.48550/arXiv.cmp-lg/9805011.

[10] Hovy, E. H.—Lin, C. Y.: Automated Multilingual Text Summarization and Its Evaluation. Technical Report. Information Sciences Institute, University of Southern California, 1999.

[11] Baeza-Yates, R.—Ribeiro-Neto, B.: Modern Information Retrieval. 2nd Edition. Addison-Wesley Publishing Company, 2008.

[12] Nenkova, A.—Vanderwende, L.—McKeown, K.: A Compositional Context Sensitive Multi-Document Summarizer: Exploring the Factors That Influence Summarization. Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06), 2006, pp. 573–580, doi: 10.1145/1148170.1148269.

[13] Nenkova, A.—Vanderwende, L.: The Impact of Frequency on Summarization. Technical Report No. MSR-TR-2005-101, Microsoft Research, Redmond, Washington, 2005.

[14] Darling, W. M.: Multi-Document Summarization from First Principles. Proceedings of the Third Text Analysis Conference (TAC 2010), 2010, `https://tac.nist.gov/publications/2010/participant.papers/Guelph_NLP.proceedings.pdf`.

[15] Carbonell, J.—Goldstein, J.: The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98), 1998, pp. 335–336, doi: 10.1145/3130348.3130369.

[16] Zhang, P. Y.—Li, C. H.: Automatic Text Summarization Based on Sentences Clustering and Extraction. 2009 2nd IEEE International Conference on Computer Science and Information Technology, 2009, pp. 167–170, doi: 10.1109/ICCSIT.2009.5234971.

[17] Haghighi, A.—Vanderwende, L.: Exploring Content Models for Multi-Document Summarization. In: Ostendorf, M., Collins, M., Narayanan, S., Oard, D. W., Vanderwende, L. (Eds.): Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2009). 2009, pp. 362–370, `https://aclanthology.org/N09-1041.pdf`.

[18] Lin, H.—Bilmes, J.: A Class of Submodular Functions for Document Summarization. In: Lin, D., Matsumoto, Y., Mihalcea, R. (Eds.): Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011). 2011, pp. 510–520, `https://aclanthology.org/P11-1052.pdf`.

[19] Stewart, G. W.: On the Early History of the Singular Value Decomposition. SIAM Review, Vol. 35, 1993, No. 4, pp. 551–566, doi: 10.1137/1035134.

[20] Golub, G. H.—Van Loan, C. F.: Matrix Computations. The Johns Hopkins University Press, 2013.

[21] Gong, Y.—Liu, X.: Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis. Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '01), 2001, pp. 19–25, doi: 10.1145/383952.383955.

[22] Steinberger, J.—Ježek, K.: Using Latent Semantic Analysis in Text Summarization and Summary Evaluation. Proceedings of the 7th Conference on Information Systems Implementation and Modeling (ISIM '04), 2004, `https://www.kiv.zcu.cz/~jstein/publikace/isim2004.pdf`.

[23] Steinberger, J.—Ježek, K.: Update Summarization Based on Novel Topic Distribution. Proceedings of the 9th ACM Symposium on Document Engineering (DocEng '09), 2009, pp. 205–213, doi: 10.1145/1600193.1600239.

[24] Paatero, P.—Tapper, U.: Positive Matrix Factorization: A Non-Negative Factor Model with Optimal Utilization of Error Estimates of Data Values. Environmetrics, Vol. 5, 1994, No. 2, pp. 111–126, doi: 10.1002/env.3170050203.

[25] Lee, J. H.—Park, S.—Ahn, C. M.—Kim, D.: Automatic Generic Document Summarization Based on Non-Negative Matrix Factorization. Information Processing & Management, Vol. 45, 2009, No. 1, pp. 20–34, doi: 10.1016/j.ipm.2008.06.002.

[26] Park, S.—Lee, J. H.—Ahn, C. M.—Hong, J. S.—Chun, S. J.: Query Based Summarization Using Non-Negative Matrix Factorization. In: Gabrys, B.,

Howlett, R. J., Jain, L. C. (Eds.): Knowledge-Based and Intelligent Information and Engineering Systems (KES 2006). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 4253, 2006, pp. 84–89, doi: 10.1007/11893011_11.

[27] Newman, M. E. J.: The Structure and Function of Complex Networks. SIAM Review, Vol. 45, 2003, No. 2, pp. 167–256, doi: 10.1137/S003614450342480.

[28] Costa, L. d. F.—Rodrigues, F. A.—Travieso, G.—Villas Boas, P. R.: Characterization of Complex Networks: A Survey of Measurements. Advances in Physics, Vol. 56, 2007, No. 1, pp. 167–242, doi: 10.1080/00018730601170527.

[29] Mihalcea, R.—Tarau, P.: TextRank: Bringing Order into Text. In: Lin, D., Wu, D. (Eds.): Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004). ACL, 2004, pp. 404–411, `https://aclanthology.org/W04-3252.pdf`.

[30] Erkan, G.—Radev, D. R.: LexRank: Graph-Based Lexical Centrality as Salience in Text Summarization. Journal of Artificial Intelligence Research, Vol. 22, 2004, No. 1, pp. 457–479, doi: 10.1613/jair.1523.

[31] Page, L.—Brin, S.—Motwani, R.—Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab, 1999, `http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf`.

[32] Bonacich, P.: Factoring and Weighting Approaches to Status Scores and Clique Identification. The Journal of Mathematical Sociology, Vol. 2, 1972, No. 1, pp. 113–120, doi: 10.1080/0022250X.1972.9989806.

[33] Otterbacher, J.—Erkan, G.—Radev, D.: Using Random Walks for Question-Focused Sentence Retrieval. In: Mooney, R., Brew, C., Chien, L. F., Kirchhoff, K. (Eds.): Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP 2005). ACL, 2005, pp. 915–922, `https://aclanthology.org/H05-1115.pdf`.

[34] Wan, X.—Yang, J.: Improved Affinity Graph Based Multi-Document Summarization. In: Moore, R. C., Bilmes, J., Chu-Carroll, J., Sanderson, M. (Eds.): Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers. ACL, 2006, pp. 181–184, `https://aclanthology.org/N06-2046.pdf`.

[35] Blei, D. M.—Ng, A. Y.—Jordan, M. I.: Latent Dirichlet Allocation. Journal of Machine Learning Research, Vol. 3, 2003, pp. 993–1022, `https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf`.

[36] Chang, Y. L.—Chien, J. T.: Latent Dirichlet Learning for Document Summarization. 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, 2009, pp. 1689–1692, doi: 10.1109/ICASSP.2009.4959927.

[37] Bengio, Y.—Ducharme, R.—Vincent, P.: A Neural Probabilistic Language Model. In: Leen, T., Dietterich, T., Tresp, V. (Eds.): Advances in Neural Information Processing Systems 13 (NIPS 2000). MIT Press, 2000, pp. 932–938, `https://proceedings.neurips.cc/paper_files/paper/2000/file/728f206c2a01bf572b5940d7d9a8fa4c-Paper.pdf`.

[38] Mikolov, T.—Sutskever, I.—Chen, K.—Corrado, G. S.—Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality. In: Burges, C. J., Bottou, L., Welling, M., Ghahramani, Z., Q., W. K. (Eds.): Ad-

vances in Neural Information Processing Systems 26 (NIPS 2013). Curran Associates, Inc., 2013, pp. 3111–3119, https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

[39] Pennington, J.—Socher, R.—Manning, C.: GloVe: Global Vectors for Word Representation. In: Moschitti, A., Pang, B., Daelemans, W. (Eds.): Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014). ACL, 2014, pp. 1532–1543, doi: 10.3115/v1/D14-1162.

[40] Joulin, A.—Grave, E.—Bojanowski, P.—Mikolov, T.: Bag of Tricks for Efficient Text Classification. In: Lapata, M., Blunsom, P., Koller, A. (Eds.): Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. ACL, 2016, pp. 427–431, https://aclanthology.org/E17-2068.pdf.

[41] Collobert, R.—Weston, J.: A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. Proceedings of the 25th International Conference on Machine Learning (ICML '08), 2008, pp. 160–167, doi: 10.1145/1390156.1390177.

[42] Levy, O.—Goldberg, Y.—Dagan, I.: Improving Distributional Similarity with Lessons Learned from Word Embeddings. Transactions of the Association for Computational Linguistics, Vol. 3, 2015, pp. 211–225, doi: 10.1162/tacl_a_00134.

[43] Liu, J.—Liu, Z.—Chen, H.: Revisit Word Embeddings with Semantic Lexicons for Modeling Lexical Contrast. 2017 IEEE International Conference on Big Knowledge (ICBK), 2017, pp. 72–79, doi: 10.1109/ICBK.2017.35.

[44] Kågebäck, M.—Mogren, O.—Tahmasebi, N.—Dubhashi, D.: Extractive Summarization Using Continuous Vector Space Models. In: Allauzen, A., Bernardi, R., Grefenstette, E., Larochelle, H., Manning, C., Yih, S. W. T. (Eds.): Proceedings of the 2nd Workshop on Continuous Vector Space Models and Their Compositionality (CVSC). ACL, 2014, pp. 31–39, doi: 10.3115/v1/W14-1504.

[45] Rossiello, G.—Basile, P.—Semeraro, G.: Centroid-Based Text Summarization Through Compositionality of Word Embeddings. In: Giannakopoulos, G., Lloret, E., Conroy, J. M., Steinberger, J., Litvak, M., Rankel, P., Favre, B. (Eds.): Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres. ACL, 2017, pp. 12–21, doi: 10.18653/v1/W17-1003.

[46] Mohd, M.—Jan, R.—Shah, M.: Text Document Summarization Using Word Embedding. Expert Systems with Applications, Vol. 143, 2020, Art. No. 112958, doi: 10.1016/j.eswa.2019.112958.

[47] Hailu, T. T.—Yu, J.—Fantaye, T. G.: A Framework for Word Embedding Based Automatic Text Summarization and Evaluation. Information, Vol. 11, 2020, No. 2, Art. No. 78, doi: 10.3390/info11020078.

[48] Saranyamol, C. S.—Sindhu, L.: A Survey on Automatic Text Summarization. International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 5, 2014, No. 6, pp. 7889–7893, https://www.ijcsit.com/docs/Volume5/vol5issue06/ijcsit20140506219.pdf.

[49] Rush, J. E.—Salvador, R.—Zamora, A.: Automatic Abstracting and Indexing. II. Production of Indicative Abstracts by Application of Contextual Inference and

Syntactic Coherence Criteria. Journal of the American Society for Information Science, Vol. 22, 1971, No. 4, pp. 260–274, doi: 10.1002/asi.4630220405.

[50] Wu, C. W.—Liu, C. L.: Ontology-Based Text Summarization for Business News Articles. Proceedings of the ISCA Eighteenth International Conference on Computers and Their Applications (CATA 2003), 2003, pp. 389–392.

[51] Canhasi, E.—Kononenko, I.: Semantic Role Frames Graph-Based Multidocument Summarization. Slovenian KDD Conference on Data Mining and Data Warehouses (SiKDD '11), 2011, doi: 10.13140/RG.2.1.2973.0646.

[52] Zwaan, R. A.—Langston, M. C.—Graesser, A. C.: The Construction of Situation Models in Narrative Comprehension: An Event-Indexing Model. Psychological Science, Vol. 6, 1995, No. 5, pp. 292–297, doi: 10.1111/j.1467-9280.1995.tb00513.x.

[53] Reeve, L.—Han, H.—Brooks, A. D.: BioChain: Lexical Chaining Methods for Biomedical Text Summarization. Proceedings of the 2006 ACM Symposium on Applied Computing (SAC '06), 2006, pp. 180–184, doi: 10.1145/1141277.1141317.

[54] Morris, J.—Hirst, G.: Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text. Computational Linguistics, Vol. 17, 1991, No. 1, pp. 21–48, `https://aclanthology.org/J91-1002.pdf`.

[55] Miller, G. A.: WordNet: A Lexical Database for English. Communications of the ACM, Vol. 38, 1995, No. 11, pp. 39–41, doi: 10.1145/219717.219748.

[56] Stone, M.: Cross-Validation: A Review. Statistics: A Journal of Theoretical and Applied Statistics, Vol. 9, 1978, No. 1, pp. 127–139, doi: 10.1080/02331887808801414.

[57] Bishop, C. M.: Pattern Recognition and Machine Learning. Springer, 2006.

[58] Kupiec, J.—Pedersen, J.—Chen, F.: A Trainable Document Summarizer. Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95), 1995, pp. 68–73, doi: 10.1145/215206.215333.

[59] Conroy, J. M.—O'leary, D. P.: Text Summarization via Hidden Markov Models. Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '01), 2001, pp. 406–407, doi: 10.1145/383952.384042.

[60] Osborne, M.: Using Maximum Entropy for Sentence Extraction. Proceedings of the ACL-02 Workshop on Automatic Summarization – Volume 4 (AS '02), 2002, pp. 1–8, doi: 10.3115/1118162.1118163.

[61] Svore, K.—Vanderwende, L.—Burges, C.: Enhancing Single-Document Summarization by Combining RankNet and Third-Party Sources. Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007, pp. 448–457, `https://aclanthology.org/D07-1047.pdf`.

[62] Burges, C.—Shaked, T.—Renshaw, E.—Lazier, A.—Deeds, M.—Hamilton, N.—Hullender, G.: Learning to Rank Using Gradient Descent. Proceedings of the 22nd International Conference on Machine Learning (ICML '05), 2005, pp. 89–96, doi: 10.1145/1102351.1102363.

[63] Jain, A.—Bhatia, D.—Thakur, M. K.: Extractive Text Summarization Using Word Vector Embedding. 2017 International Conference on Machine Learning and

Data Science (MLDS), 2017, pp. 51–55, doi: 10.1109/MLDS.2017.12.

[64] Cao, Z.—Li, W.—Li, S.—Wei, F.—Li, Y.: AttSum: Joint Learning of Focusing and Summarization with Neural Attention. In: Matsumoto, Y., Prasad, R. (Eds.): Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. 2016, pp. 547–556, `https://aclanthology.org/C16-1053.pdf`.

[65] Nallapati, R.—Zhai, F.—Zhou, B.: SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 31, 2017, No. 1, pp. 3075–3081, doi: 10.1609/aaai.v31i1.10958.

[66] Zhou, Q.—Yang, N.—Wei, F.—Huang, S.—Zhou, M.—Zhao, T.: Neural Document Summarization by Jointly Learning to Score and Select Sentences. In: Gurevych, I., Miyao, Y. (Eds.): Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). ACL, 2018, pp. 654–663, doi: 10.18653/v1/P18-1061.

[67] Liu, Y.—Lapata, M.: Text Summarization with Pretrained Encoders. In: Inui, K., Jiang, J., Ng, V., Wan, X. (Eds.): Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). ACL, 2019, pp. 3730–3740, doi: 10.18653/v1/D19-1387.

[68] Sutton, R. S.—Barto, A. G.: Reinforcement Learning: An Introduction. MIT Press, 2018.

[69] Puterman, M. L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, 2014.

[70] Ryang, S.—Abekawa, T.: Framework of Automatic Text Summarization Using Reinforcement Learning. In: Tsujii, J., Henderson, J., Paşca, M. (Eds.): Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP 2012). ACL, 2012, pp. 256–265, `https://aclanthology.org/D12-1024.pdf`.

[71] Rioux, C.—Hasan, S. A.—Chali, Y.: Fear the REAPER: A System for Automatic Multi-Document Summarization with Reinforcement Learning. In: Moschitti, A., Pang, B., Daelemans, W. (Eds.): Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014). ACL, 2014, pp. 681–690, doi: 10.3115/v1/D14-1075.

[72] Genest, P. E.—Lapalme, G.: Fully Abstractive Approach to Guided Summarization. In: Li, H., Lin, C. Y., Osborne, M., Lee, G. G., Park, J. C. (Eds.): Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers. ACL, 2012, pp. 354–358, `https://aclanthology.org/P12-2069.pdf`.

[73] Harabagiu, S. M.—Lacatusu, F.: Generating Single and Multi-Document Summaries with GISTEXTER. Document Understanding Conferences, 2002, `https://www-nlpir.nist.gov/projects/duc/pubs/2002papers/utdallas_sanda.pdf`.

[74] Lee, C. S.—Jian, Z. W.—Huang, L. K.: A Fuzzy Ontology and Its Application to News Summarization. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol. 35, 2005, No. 5, pp. 859–880, doi: 10.1109/TSMCB.2005.845032.

[75] Genest, P. E.—Lapalme, G.: Framework for Abstractive Summarization Using Text-to-Text Generation. In: Filippova, K., Wan, S. (Eds.): Proceedings of the Workshop on Monolingual Text-to-Text Generation. ACL, 2011, pp. 64–73, `https://aclanthology.org/W11-1608.pdf`.

[76] Greenbacker, C.: Towards a Framework for Abstractive Summarization of Multimodal Documents. In: Petrovic, S., Selfridge, E., Pitler, E., Osborne, M., Solorio, T. (Eds.): Proceedings of the ACL 2011 Student Session. ACL, 2011, pp. 75–80, `https://aclanthology.org/W11-1608.pdf`.

[77] Joshi, A. K.—Schabes, Y.: Tree-Adjoining Grammars. Handbook of Formal Languages: Volume 3 Beyond Words, Springer, 1997, pp. 69–123.

[78] McDonald, D. D.—Greenbacker, C.: 'If You've Heard It, You Can Say It' – Towards an Account of Expressibility. In: Kelleher, J., Namee, B. M., van der Sluis, I. (Eds.): Proceedings of the 6th International Natural Language Generation Conference (INLG 2010). ACL, 2010, `https://aclanthology.org/W10-4220.pdf`.

[79] Moawad, I. F.—Aref, M.: Semantic Graph Reduction Approach for Abstractive Text Summarization. 2012 Seventh International Conference on Computer Engineering & Systems (ICCES), 2012, pp. 132–138, doi: 10.1109/ICCES.2012.6408498.

[80] Jordan, M. I.: Serial Order: A Parallel Distributed Processing Approach. Technical Report, June 1985–March 1986. Technical Report. California University, San Diego, La Jolla (USA), Institute for Cognitive Science, 1986.

[81] Bengio, Y.—Frasconi, P.—Simard, P.: The Problem of Learning Long-Term Dependencies in Recurrent Networks. IEEE International Conference on Neural Networks, Vol. 3, 1993, pp. 1183–1188, doi: 10.1109/ICNN.1993.298725.

[82] Hochreiter, S.—Schmidhuber, J.: Long Short-Term Memory. Neural Computation, Vol. 9, 1997, No. 8, pp. 1735–1780, doi: 10.1162/neco.1997.9.8.1735.

[83] Cho, K.—Van Merriënboer, B.—Gulcehre, C.—Bahdanau, D.—Bougares, F.—Schwenk, H.—Bengio, Y.: Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In: Moschitti, A., Pang, B., Daelemans, W. (Eds.): Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014). ACL, 2014, pp. 1724–1734, doi: 10.3115/v1/D14-1179.

[84] Sutskever, I.—Vinyals, O.—Le, Q. V.: Sequence to Sequence Learning with Neural Networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K. Q. (Eds.): Advances in Neural Information Processing Systems 27 (NIPS 2014). Curran Associates, Inc., 2014, pp. 3104–3112, `https://proceedings.neurips.cc/paper_files/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf`.

[85] Bahdanau, D.—Cho, K.—Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate. CoRR, 2014, doi: 10.48550/arXiv.1409.0473.

[86] Vaswani, A.—Shazeer, N.—Parmar, N.—Uszkoreit, J.—Jones, L.—Gomez, A. N.—Kaiser, Ł.—Polosukhin, I.: Attention Is All You Need. In: Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017). Curran Associates, Inc., 2017, pp. 5998–6008, doi: 10.48550/arXiv.1706.03762.

[87] Li, J.—Tang, T.—Zhao, W. X.—Nie, J. Y.—Wen, J. R.: Pre-Trained Language Models for Text Generation: A Survey. ACM Computing Surveys, Vol. 56, 2024, No. 9, Art. No. 230, doi: 10.1145/3649449.

[88] Xu, C.—McAuley, J.: A Survey on Dynamic Neural Networks for Natural Language Processing. In: Vlachos, A., Augenstein, I. (Eds.): Findings of the Association for Computational Linguistics: EACL 2023. ACL, 2023, pp. 2370–2381, doi: 10.18653/v1/2023.findings-eacl.180.

[89] Clarke, J.—Lapata, M.: Models for Sentence Compression: A Comparison Across Domains, Training Requirements and Evaluation Measures. In: Calzolari, N., Cardie, C., Isabelle, P. (Eds.): Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006). ACL, 2006, pp. 377–384, doi: 10.3115/1220175.1220223.

[90] Hori, C.—Furui, S.: Speech Summarization: An Approach Through Word Extraction and a Method for Evaluation. IEICE TRANSACTIONS on Information and Systems, Vol. 87, 2004, No. 1, pp. 15–25.

[91] Filippova, K.: Multi-Sentence Compression: Finding Shortest Paths in Word Graphs. In: Huang, C. R., Jurafsky, D. (Eds.): Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010). ACL, 2010, pp. 322–330, https://aclanthology.org/C10-1037.pdf.

[92] Jing, H.: Sentence Reduction for Automatic Text Summarization. Sixth Applied Natural Language Processing Conference (ANLP-NAACL 2000), ACL, 2000, pp. 310–315, doi: 10.3115/974147.974190.

[93] Knight, K.—Marcu, D.: Statistics-Based Summarization – Step One: Sentence Compression. AAAI/IAAI, Vol. 2000, 2000, pp. 703–710, https://api.semanticscholar.org/CorpusID:9363872.

[94] Cheng, J.—Lapata, M.: Neural Summarization by Extracting Sentences and Words. In: Erk, K., Smith, N. A. (Eds.): Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). ACL, 2016, pp. 484–494, doi: 10.18653/v1/P16-1046.

[95] Zhang, X.—Lapata, M.—Wei, F.—Zhou, M.: Neural Latent Extractive Document Summarization. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (Eds.): Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). ACL, 2018, pp. 779–784, doi: 10.18653/v1/D18-1088.

[96] Jing, H.—McKeown, K.: Cut and Paste Based Text Summarization. 1st Meeting of the North American Chapter of the Association for Computational Linguistics (ANLP 2000), 2000, https://aclanthology.org/A00-2024.pdf.

Daniel O. Cajueiro holds his Ph.D. in electronics and computer engineering and currently works as Professor at the Department of Economics at UnB. He is also a senior researcher at LAMFO. His research comprises elements of economics, social sciences and artificial intelligence.

Arthur G. Nery holds his BA in economics and is currently Researcher at LAMFO. His research comprises elements of social sciences and machine learning.

Igor Tavares holds his BA in mechanic engineering. His research considers elements of mechanic engineering, numerical analysis and machine learning.

Maísa K. De Melo holds her Ph.D. in mathematical and computational modeling and currently works as Professor at the Department of Mathematics at IFMG. She is also Senior Researcher at LAMFO. Her research comprises elements of optimization, finance and computer science.

Silvia A. dos Reis holds her Ph.D. in production engineering and currently works as Professor at the Department of Business at UnB. She is also Senior Researcher at LAMFO. Her research comprises elements of mathematical modeling and decision support models.

Li Weigang holds his Ph.D. in computer science and is currently Professor in the Department of Computer Science at UnB. His research encompasses elements of artificial intelligence and systems management.

Victor R. R. Celestino holds his Ph.D. in psychology and his B.Sc. in aeronautical engineering. Currently he works as Professor at the Department of Administration at UnB. He is also Senior Researcher at LAMFO. His research considers elements of operations research, artificial intelligence and transport engineering.