A Long Texts Summarization Approach to Scientific Articles

Cinthia M. Souza[1], **Renato Vimieiro**[1]

[1]Universidade Federal de Minas Gerais, Brazil

{cinthiasouza, rvimieiro}@dcc.ufmg.br

Abstract. *Automatic text summarization aims at condensing the contents of a text into a simple and descriptive summary. Summarization techniques drastically benefited from the recent advances in Deep Learning. Nevertheless, these techniques are still unable to properly deal with long texts. In this work, we investigate whether the combination of summaries extracted from multiple sections of long scientific texts may enhance the quality of the summary for the whole document. We conduct experiments on a real world corpus to assess the effectiveness of our proposal. The results show that our multi-section proposal is as good as summaries generated using the entire text as input and twice as good as single section.*

1. Introduction

In this work we propose an alternate approach for summarizing long scientific texts. We investigate whether the combination of summaries extracted from multiple sections may enhance the quality of the summary for the whole document. This could be of particular interest for summarization methods that are unable to deal with long texts due to size restrictions, particularly those with high computing demands.

Automatic text summarization aims at facilitating access to information. The objective of this technique is to condense the information in a text into a simple and descriptive summary, which gives the reader a general idea of the text without having to read its entire content. Text summarization techniques can be divided into two groups: extractive; and abstractive. Extractive Text Summarization (ETS) techniques assign a score to each sentence of the text and select n sentences with the highest score to compose the summary. Abstractive Text Summarization (ATS) techniques are trained to generate a natural language summary using an internal representation of the text. Because they have a large dictionary, these techniques do not necessarily create summaries with the same words as the text. This makes the summarization process more similar to humans'. For a long time, ETS techniques used simpler models of unsupervised learning and ATS techniques were little explored compared to ETS. In recent years, ETS and ATS techniques have had great improvements due to the use of Deep Learning (DL) models.

Most of recent works explore text summarization in news corpora such as CNN-Daily/Mail, DUC Corpus and Gigaword. These corpora are mostly comprised of short texts with approximately 650 words [Nallapati et al., 2016]. In short texts, DL models are easily applicable and perform well compared to unsupervised techniques. However, DL models present some challenges regarding their training, such as the difficulty of working with long texts, due to the high computational cost [Ding et al., 2020]. Solutions to this problem are to use sliding windows, simplify the architecture, or set a maximum number of words that will be used as input. There are different negative implications attached to

using each of these strategies. One of them is the loss of important information. In the task of scientific text summarization, for example, the abstracts of the articles, which are used as ground-truth, are created by experts, and follow a standardized structure, in which there is a contextualization, the problem, the solution and the evaluation of results. This content is commonly distributed in different sections of the text. So not considering all the information in the text reduces the solution space, resulting in a loss of performance. We compare five ETS algorithms in a long text dataset[1], composed of scientific articles, published by the Plos One journal. Our main contributions are:

- Assessing the performance of ETS methods on a long text corpus;
- Exploring the segmentation of scientific articles considering the structural pattern of scientific abstracts;
- Evaluating the contribution of the sections in the generation of summaries;
- Evaluating the impact of combining multi-section summaries on the final performance of algorithms;
- Validating the results using the set of metrics Recall-Oriented Understudy for Gisting Evaluation[2] (ROUGE) [Lin, 2004].

The remaining of this manuscript is organized as follows. In Section 2 we present a perspective of how the summarization task has been explored. Then, in Section 3 we describe the algorithms used in this work. Sections 4 and 5 contain the proposed approach with a description of the corpus and the methodological steps and results obtained, respectively. Finally, we conclude the manuscript presenting our final remarks and future works in Section 6.

2. Related Works

ETS techniques are more suitable to particular tasks than ATS techniques. For instance, in situations where it is mandatory to have full control of the content present in the summaries, like in the summarization of scientific and legal documents, ETS is more apropriate since changing or introducing more sentences to the summary may alter the document meaning. We discuss in this section some recent works in the area of ETS. These works were chosen in order to present a perspective of how the ETS task is currently being explored.

Gidiotis and Tsoumakas [2020] proposed in their work a divide-and-conquer approach for long text summarization. The proposed approach uses discourse structure and sentence similarity to create a dataset composed of pairs of short texts and their summaries. The aim of the authors is to reduce the complexity of the problem, consequently, reducing the computational cost. The proposed method was tested on different summarization algorithms, including SumBasic [Vanderwende et al., 2007], LexRank [Erkan and Radev, 2004], and PEGASUS [Zhang et al., 2020]. According to the authors, the best models obtained presented results that were competitive with the state-of-the-art (SOTA). Dong et al. [2021] proposes an unsupervised ETS model for long scientific texts based on graphs. Their approach works on a hierarchical graph representing the document. The hierarchy has two levels of connection, intra-section and inter-section. The similarity between sentences is calculated using the cosine-similarity and the importance of each

[1]Code and dataset available at: https://github.com/CinthiaS/long_text_summarization

[2]Available at: https://github.com/chakki-works/sumeval

sentence is the sum of the intra and inter-section importance. The intra-section is the comparison of sentences within the same section. The inter-section is obtained by comparing the sentences of a section with those of other topics/sections of the document. The summaries are created by extracting the sentences with the highest score. The experiments were conducted using PubMed and ArXiv datasets. The results were compared with supervised and unsupervised models, in addition to the baseline lead, which selects the first k tokens as the summary, and an Oracle. To validate the results, the metrics ROUGE-1, ROUGE-2, ROUGE-L and the human evaluation were used. The results obtained were better than all the compared algorithms.

Among the works studied, only Gidiotis and Tsoumakas [2020] explores text segmentation as a strategy for long text summarization. The results obtained by the authors show that the summarization of sections of the text, individually, and the creation of summaries as the composition of these summaries is an efficient strategy for long text summarization. Differently from the proposal of this work, Gidiotis and Tsoumakas [2020] propose an agnostic strategy to the knowledge domain of the text. However, we believe that performing a segmentation considering the abstract structure of articles can achieve better results. Furthermore, based on the works of Xu et al. [2019], Zhong et al. [2020], Xiao et al. [2020], and Zhang et al. [2020], which are important references in the field, it is possible to see that these, in general, focus on news corpora, which are characterized by short text and summaries. Thus, there is a need to explore strategies that can allow the use of reference models in the area in this scenario.

3. Extractive Summarization Algorithms

In this work, five ETS algorithms are tested: SumBasic; LexRank; TextRank; and two models based on Bidirectional Encoder Representations from Transformers (BERT). SumBasic was selected as the baseline method for comparison. LexRank and TextRank are graph-based algorithms that stand out for having a simple approach with competitive results. The last two algorithms are approaches that use latest SOTA natural language processing components, which are language models created with the BERT architecture. SumBasic [Vanderwende et al., 2007] evaluates the importance of the words in the text based on their frequencies. After, it assigns an importance to the sentences of the text according to the importance of their words [Vanderwende et al., 2007]. The idea is that the more important words a sentence has, the more important it is. LexRank [Erkan and Radev, 2004] and TextRank [Mihalcea and Tarau, 2004] are graph-based algorithms. Basically, these algorithms create a representation of the text in a weighted undirected graph, where vertices are sentences, edges represent the relationship between two sentences, and edge weights are the similarity between them. The main differences between these algorithms is the calculation of similarity. LexRank defines the similarity between two sentences as the cosine of their Term Frequency–Inverse Document Frequency (TF-IDF) vector representations, while TextRank uses a measure of overlap between the words in the text, normalized by the length of the sentences. After creating the text representation, both algorithms use the PageRank algorithm [Page et al., 1999] to assign a score to the sentences. The summary is composed of the k sentences with the highest score.

The BERT Summarizer algorithm is based on clustering and uses representations of embeddings generated by the BERT model. BERT is a pre-trained Transformer architecture [Vaswani et al., 2017], designed for creating deep representations of unlabeled

text. One of the advantages of BERT is that the architecture used is bidirectional, making it possible to associate forward and backward contexts for all layers, unlike, for example, the Generative Pre-trained Transformer (OpenAI GPT), which is unidirectional [Devlin et al., 2018]. Devlin et al. [2018] describes BERT as being conceptually simple and empirically powerful. Basically, the BERT Summarizer obtains the embedding representations of the text sentences using BERT and generates a matrix where lines represent sentences and columns represent the dimensionality of the embedding vector. This matrix is used as input for the K-means algorithm, together with the number k of clusters which also represents the number of sentences to be extracted. At the end, the sentence that has the smallest distance from the centroid is added to the summary.

4. Proposed Approach

In this work we use a corpus of scientific articles published by Plos One. The corpus is publicly available from the journal's website[3]. The collected articles were segmented considering their division of sections. To retrieve these sections, it is important to use a tagged base that allows the recognition of these sections. The data provided by Plos One is made available in XML, allowing easy recognition of sections. The experiments are carried out in a dataset with 5000 articles, selected at random. Each document is segmented into four sections. The segmentation of the dataset is used as a strategy to work with long texts, reducing the amount of data in the input of the algorithm and enabling the capture of information from each section in order to generate comprehensive summaries, covering each topic of the abstract. Table 1 presents the name of the sections extracted from the articles, the acronyms used to identify them, the average number of sentences and words in each of them and the compression ratio, which is the ratio between the number of sentences in the section and the number of sentences in the abstract.

Table 1. Description of the corpus used in the experiments

Section	Acronyms	Average number of sentences	Average number of words	Compression
Abstract	S_1	11	210	-
Introduction	S_2	23	540	2,09
Materials	S_3	59	1077	5,36
Results and Conclusion	S_4	110	2081	10,00
All document	D_m	192	3698	17,45

Our approach is divided into three steps. In the first step, data collection and preprocessing is performed. The dataset creation process consists of three phases: collection, refactoring and segmentation. Initially, the documents are collected in XML format. In some cases, the XML document did not have a tag delimiting the text sections. By consequence, the collected documents were refactored in order to correct inconsistencies and facilitate the segmentation process. The refactoring process generated a new XML document where all target sections are properly tagged. The documents are segmented afterwards into four sections, S_1, S_2, S_3, S_4. The section S_1 is the abstract of the article, used as ground-truth, henceforth called the reference summary. The preprocessing step follows the document segmentation. The first task in this step is removal of text citations and section titles. After, the XML is converted to text and noise, that is special characters,

[3]Available at: http://api.plos.org/text-and-data-mining/ - Accessed on: Aug 2021

excess spaces and line breaks, and unicode symbols are removed. Finally, all texts are lowercased, stop words are removed, and the words remaining are stemmed.

In the second step, the texts are summarized using the five algorithms described in Section 3. Initially, the experiments are performed by section. Each algorithm receives the text of each section, separately, and generates a summary of each. The algorithms used receive the number k of sentences to be extracted. We define that the summaries of each section are generated with $k = 3$, so the composition of the summaries will have 9 sentences which is, approximately, the average number of sentences in an abstract (see Table 1). Subsequently, a summary is generated from the entire content of the text. The objective of these experiments is: (1) evaluate the contribution of each section to the summary and (2) evaluate whether the summaries created in each section, separately, have as good results as using the entire text as input. Two summary generation approaches were used. The first, called A_1, uses all the text as input to the algorithms. The second, called A_2, creates a summary for each section and combines the summaries. A_1 uses $k = 9$ and A_2 uses $k = 3$, thus, A_2 generates a summary with 9 sentences, 3 from each section. Thus, both are limited to the same summary size. The SumBasic, LexRank and TextRank algorithms are unsupervised, so they do not require training. Models using BERT were implemented with an API[4] developed by Miller [2019]. The difference between the two algorithms is that, the first one, called BERT Basic uses a generic pre-trained model, provided by the organization Hugging Face[5]. The second one, called SciBERT Summ, uses a pre-trained model created from scientific texts, provided by the Allen Institute for AI, called SciBERT[6].

Finally, the third step is the evaluation of the generated summaries. For this, the metrics ROUGE-1 (R1), ROUGE-2 (R2) and ROUGE-L (RL) are used. The ROUGE metrics is widely used in the literature. ROUGE-N evaluates the overlap of n-grams between the candidate and reference summary [Sanchez-Gomez et al., 2018]. ROUGE-L evaluates the correspondence between the Longest Common Substring (LCS) shared by two sentences [Sanchez-Gomez et al., 2018]. Both metrics assign a score from 0 to 1 to each sentence, where 0 represents that the candidate summary does not capture any information from the reference summary and 1 represents that the candidate summary captures all information of the reference summary. The results obtained with the summarization algorithms, which we called candidate summary, are compared with the reference summaries using ROUGE metrics. The performance of an algorithm is calculated as the average of the metrics for each generated summary.

5. Results

Initially, the summary of each section of the text was performed using the algorithms presented in Section 4. For each section, 3 sentences were extracted. The results obtained were compared with the reference summary and are presented in Table 2a. The first column presents the name of the algorithm used, the second presents the acronyms of the section (see Table 1). The columns R1, R2 and RL show the results of the metrics and, the last column, presents the average of the three metrics. The algorithm that presented the best result is highlighted.

[4]https://github.com/dmmiller612/bert-extractive-summarizer

[5]https://github.com/huggingface/transformers

[6]https://github.com/allenai/scibert

Table 2. Results of ETS experiments conducted in the Introduction (S_2), Materials and Methods (S_3), and Results and Conclusion (S_4) sections and using as input all text (A_1) and using a summary created from the combination of multi-section summaries (A_2).

(a) Results by section

		R1 (%)	R2 (%)	RL (%)	Average
SumBasic	S_2	15.24	4.78	10.52	10.2
	S_3	10.30	3.64	7.89	7.3
	S_4	18.25	6.67	12.68	12.5
LexRank	S_2	17.36	5.55	11.69	11.5
	S_3	13.10	4.84	9.78	9.2
	S_4	25.60	10.01	16.75	17.5
TextRank	S_2	19.76	6.23	12.92	**13.0**
	S_3	15.35	5.51	11.02	**10.6**
	S_4	27.78	10.57	17.74	**18.7**
BERT Basic	S_2	15.45	5.12	10.94	10.5
	S_3	10.91	4.61	8.69	8.1
	S_4	20.21	8.27	14.46	14.3
SciBERT Summ	S_2	15.57	5.16	11.00	10.6
	S_3	11.29	4.69	8.91	8.3
	S_4	21.06	8.68	14.93	14.9

(b) Results by approach

		R1 (%)	R2 (%)	RL (%)	Average
SumBasic	A_1	31.62	10.21	17.27	19.7
	A_2	28.15	10.18	17.37	18.6
LexRank	A_1	37.36	14.44	20.69	**24.2**
	A_2	33.22	12.88	20.05	**22.1**
TextRank	A_1	33.17	13.09	18.80	21.7
	A_2	33.01	12.88	19.69	21.9
BERT Basic	A_1	30.70	10.07	17.34	19.37
	A_2	29.39	11.97	18.97	20.1
SciBERT Summ	A_1	31.53	10.82	17.59	19.98
	A_2	29.79	12.17	19.07	20.3

By comparing the scores present in Table 2a, we concluded that the worst results were obtained using the section Materials and Methods (S_3) and the best ones were using Results and Conclusion (S_4). For the TextRank algorithm, for example, which showed the best performance, the difference between the average of the metrics for the results with Introduction (S_2) was 5.7% and 8.1% for S_3. In all cases, the average results of BERT Basic and SciBERT Summ were worse compared to TextRank and LexRank, being superior only to the baseline, SumBasic. Thus, we concluded that the text sections present different degrees of contribution to the summary generation. The choice of the sections in which the experiments are performed can generate significant changes in the results of the algorithms. Furthermore, it is possible to verify that, although S_2 and S_3 have a lower performance than S_4, there is a contribution of these sections in the summary. After, was performed a comparison between two summarization approaches. Table 2b presents the results obtained with these experiments. The first column presents the algorithms used, the second column identifies the approach used, the columns R1, R2 and RL present the values of the metrics in percentage and, the last column, presents the average of the three metrics.

From Table 2b, it is possible to verify that the results using the combination of summaries is twice as good that summaries of only one section. Thus, it can be concluded that the combination of multi-section summaries is capable of producing high quality summaries. The difference between the approaches is 1.10% for SumBasic, 2.10% for LexRank, 0.2% for TextRank, for BERT Basic, 0.73% and 0.32% for SciBERT Summ. Based on the results, we conclude that segmenting the texts, considering the structural pattern of the abstract, and summarize each section and combine them is a strategy that can help in the task of long texts summarization, reducing the computational cost of algorithms and can mitigate the loss of information. Among algorithms, the best result was obtained with LexRank, for both approaches. The difference between the averages of LexRank with A_1 for the other algorithms were 4.5% for SumBasic, 2.5% for TextRank, 4.38% for BERT Basic, and 4.22% for SciBERT Summ. For A_2 the differences were 3.5% for SumBasic, 0.2% for TextRank, 2.0% for BERT Basic, and 1.8% for SciBERT Summ. TextRank presented the smallest percentage difference between the approaches.

Showing that the summaries of both approaches are very similar. The SumBasic algorithm, which is used as a baseline, had the worst performance. BERT based algorithms showed a worse performance compared to LexRank and TextRank. Even though TextRank has the smallest difference between the approaches, the average metric value of A_1 and A_2 shows that LexRank performs better, with a difference of 2.5% with A_1 and 0.2% with A_2.

6. Final Considerations

Currently, text summarization has shown significant improvements due to the use of DL techniques. However, in this context, long texts summarization is still challenging. In this work, the contribution of different sections of the text in the composition of summaries of scientific articles are evaluated. The results obtained show that the text sections have different degrees of contribution in the generation of summaries. In this experiment, the algorithm with the best performance was TextRank, with a mean of the metrics of 13% for Introduction, 10.6% for Materials and Methods, and 18.7% for Results and Conclusion. Considering all algorithms the best results were obtained with the Results and Conclusion section. When comparing approaches A_1 and A_2, the biggest difference was 2.1% and a lowest was 0.2%. This demonstrated that the combination of multi-section summaries can generate summaries of similar quality compared to using the entire text as input. In this experiment, the best performance was obtained with the LexRank algorithm, with an average score of 22.1% for the proposed approach. Although the results of the metrics obtained in this work are inferior to the SOTA, we believe that the results obtained are promising, as it was conducted in a reduced dataset, using simple unsupervised algorithms and pre-training language representations without no adjustment to the knowledge domain in which it was applied. For future work, we intend to reproduce the experiments in a larger dataset, develop a specific solution for scientific texts and evaluate the results using metrics that allow evaluating using other metrics. The ROUGEs metrics are widely used in the literature, however works such as Souza et al. [2021] and Kane et al. [2020] question the performance of these metrics and highlight the need to explore other metrics.

Acknowledgements

We would like to thank the partial support from MPMG through the project Analytical Capabilities.

References

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *The 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, 2018.

M. Ding, C. Zhou, H. Yang, and J. Tang. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems*, 33, 2020.

Y. Dong, A. M. Romascanu, and J. C. K. Cheung. Discourse-aware unsupervised summarization for long scientific documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1089–1102, 2021.

G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

A. Gidiotis and G. Tsoumakas. A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:3029–3040, 2020.

H. Kane, M. Y. Kocyigit, A. Abdalla, P. Ajanoh, and M. Coulibali. Nubia: Neural based interchangeability assessor for text generation. pages 28–37. Association for Computational Linguistics, 2020.

C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

R. Mihalcea and P. Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

D. Miller. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*, 2019.

R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, page 280–290, 2016.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez. Extractive multi-document text summarization using a multi-objective artificial bee colony optimization approach. *Knowledge-Based Systems*, 159:1–8, 2018.

C. M. Souza, M. R. Meireles, and P. E. Almeida. A comparative study of abstractive and extractive summarization techniques to label subgroups on patent dataset. *Scientometrics*, 126(1):135–156, 2021.

L. Vanderwende, H. Suzuki, C. Brockett, and A. Nenkova. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618, 2007.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

D. Xiao, H. Zhang, Y. Li, Y. Sun, H. Tian, H. Wu, and H. Wang. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3997–4003, 2020.

J. Xu, Z. Gan, Y. Cheng, and J. Liu. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031. Association for Computational Linguistics, 2019.

J. Zhang, Y. Zhao, M. Saleh, and P. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 6197–6208. Association for Computational Linguistics, 2020.