ANDRÉ SEIDEL OLIVEIRA

Summarizing Multiple Websites for Automatic PT-BR Wikipedia Generation

São Paulo
2023

ANDRÉ SEIDEL OLIVEIRA

Summarizing Multiple Websites for Automatic PT-BR Wikipedia Generation

Versão Original

Dissertation presented to Escola Politécnica of Universidade de São Paulo to obtain Master of Sciences degree.

São Paulo
2023

ANDRÉ SEIDEL OLIVEIRA

Summarizing Multiple Websites for Automatic PT-BR Wikipedia Generation

Versão Original

Dissertation presented to Escola Politécnica of Universidade de São Paulo to obtain Master of Sciences degree.

Concentration area:

Computer Engineering

Advisor:

Ph.D. Anna Helena Reali Costa

São Paulo
2023

Catalogação-na-publicação

Oliveira, André Seidel
Summarizing multiple websites for automatic PT-BR Wikipedia generation / A. S. Oliveira, A. H. R. Costa -- São Paulo, 2023.
70 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Inteligência artificial 2.Processamento de linguagem natural 3.Redes neurais I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t. III.Costa, Anna Helena Reali

ACKNOWLEDGMENTS

First, I would like to express my deepest gratitude to my professor and advisor Anna Reali for her invaluable teachings and patience throughout this journey. I am also thankful for my defense committee, professors Bruno and Thiago, who generously provided their expertise.

Special thanks to my family Márcia, Fernanda, and José Teófilo, and fiancee, Débora. I could not have undertaken this journey without you. Lastly, I had the pleasure of collaborating with Lucas Francisco Pellicer and Thomas Palmeira Ferraz, that supported this research in difficult moments.

This research was carried out with the support of *Itaú Unibanco S.A.*, through the scholarship program of *Programa de Bolsas Itaú* (PBI), and it is also financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Finance Code 001, Brazil. Any opinions, findings, and conclusions expressed in this manuscript are those of the authors and do not necessarily reflect the views, official policy, or position of the Itaú-Unibanco and CAPES.

"If a machine is expected to be infallible, it cannot also be intelligent."

-- Alan M. Turing

ABSTRACT

Wikipedia is an essential free source of intelligible knowledge. Despite that, the Brazilian Portuguese portal still lacks descriptions for many subjects. To expand the Brazilian Wikipedia, we present PLSum, Portuguese Long Summarizer, a framework for generating wiki-like abstractive summaries from multiple descriptive websites. The framework has an extractive stage followed by an abstractive one. In the extractive stage, parts from documents are extracted on the topic of interest. Then in the abstractive step, fine-tuning is performed, seeking to rewrite the excerpts in a cohesive, correct, and meaningful summary. In particular, we fine-tune and compare two recent variations of the Transformer neural network for the abstractive stage, PTT5 and Longformer. In the extractive stage, we propose a new method based on clustering dense semantic representations to select the most relevant sentences. To fine-tune and evaluate the model, we created a dataset with thousands of examples, linking reference websites to Wikipedia. Our final results show that it is possible to generate meaningful abstractive summaries from Brazilian Portuguese web content. PLSum successfully applies style transfer, which is not possible with fully extractive techniques that are predominant in Brazilian literature. Finally, we also concluded that the use of dense semantic representations for the extractive stage enabled the selection of diverse sentences, making a non repetitive extractive summary.

Keywords – Abstractive Summarization, Natural Language Processing, Multi-document Summarization, Machine Learning.

RESUMO

A Wikipédia é uma importante fonte gratuita de conhecimento inteligível. Apesar disso, o portal em português do Brasil ainda carece de descrições para muitos assuntos. Em um esforço para expandir a Wikipédia brasileira, apresentamos PLSum, *Portuguese Long Summarizer*, um arcabouço para gerar resumos abstrativos no estilo da Wikipédia a partir de vários sítios (*sites*) descritivos. O arcabouço possui uma etapa extrativa seguida por uma abstrativa. Na etapa extrativa, extraem-se trechos de documentos sobre o tema de interesse e, na etapa abstrativa, é realizado um ajuste fino, buscando reescrever os trechos em um resumo coeso, correto e significativo. Em particular, para a etapa abstrativa, ajustamos e comparamos duas variações recentes da rede neural do tipo *Transformer*, a PTT5 e o Longformer. Já na etapa extrativa, inovamos ao propor um método baseado em agrupamento de representações semânticas vetoriais para seleção de sentenças relevantes. Para ajustar e avaliar os modelos, criamos uma base de dados para sumarização multi-documentos com milhares de exemplos, vinculando sítios de referência às páginas do Wikipédia. Nossos resultados mostram que é possível gerar resumos abstrativos significativos a partir do conteúdo da web em português do Brasil. Além disso, mostramos que o PLSum tem sucesso na aplicação da transferência de estilo de escrita, o que não é possível com as técnicas totalmente extrativas, predominantes na literatura. Por fim, nós também concluímos que o método de agrupamento de representações semânticas vetoriais possibilitou a extração de sentenças mais diversas na etapa extrativa.

Palavras-Chave – Sumarização Abstrativa, Processamento de Linguagem Natural, Sumarização Multi-documentos, Aprendizado de Máquina.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

BERT – Bidirectional Encoder Representations from Transformers

BERTimbau – BERT trained with Portuguese texts.

BLEU – Bilingual Evaluation Understudy

BRWac – Brazilian Portuguese Web as a Corpus

BRWac2Wiki – Brazilian Portuguese Web as a Corpus to Wikipedia

CBOW – Continuos-Bag-Of-Words

ClusterSum – Cluster Summarizer

CST – Cross-document Structure Theory

CSTNews – Cross-document Structure Theory News

CSTSumm – Cross-document Structure Theory Summarizer

DUC – Document Understanding Conference

GloVe – Global Vectors for Word Representations

GRU – Gated Recurrent Unit

InIt – Information Item

LCS – Longest Common Subsequence

LSTM – Long Short-Term Memory

MD – Multi-document

MDAS – Multi-document Abstractive Summarization

MDS – Multi-document Summarization

METEOR – Metric for Evaluation of Translation with Explicit Ordering

MMR – Maximal Marginal Relevance

MT – Machine Translation

NLP – Natural Language Processing

NN – Neural Network

NP – Noun Phrase

PAS – Predicate-Argument Structure

PG – Pointer-Generator

PG-MMR – Pointer-Generator - Maximal Marginal Relevance

PLSum – Portuguese Long Summarizer

POS – Part-of-Speech

PTT5 – Portuguese Text-to-Text Transfer Transformer

RNN – Recurrent Neural Network

ROUGE – Recall-Oriented Understudy for Gisting Evaluation

RSumm – Summarization method proposed by Rafael Ribaldo *et al.* ((RIBALDO et al., 2012)).

SBERT – Sentence Bidirectional Encoder Representations from Transformers

SCU – Summarization Content Units

Seq2Seq – Sequence-to-Sequence

SD – Single-document

SDS – Semantic Textual Simmilarity

TAC – Text Analysis Conference

T-DMCA – Transformer - Decoder with Memory-Compressed Attention

TF-IDF – Term Frequency - Inverse Document Frequency

T5 - Text-to-Text Transfer Transformer

UMAP – Uniform Manifold Approximation and Projection for Dimension Reduction

VP – Verb Phrase

Word2Vec – Word to Vector

WikiSumm - Name of summarization algorithm and dataset proposed by (LIU et al., 2018).

CONTENTS

1 INTRODUCTION

Wikipedia is a public domain encyclopedia project, being a vast source of intelligible information on many subjects. It has more than 51 million articles in over 300 languages and around 1 million written in Portuguese. Despite the tremendous voluntary effort to maintain it, many topics still need to be covered, especially in languages other than English. For example, if we assume that Wikipedia articles describe language-independent topics, a quick comparison between the current number of articles in Portuguese and English shows that Portuguese Wiki lacks at least 5 million topics[1].

In contrast, there are plenty of textual descriptions online for just about anything. In this sense, automatic text summarization techniques could be applied to these descriptions to generate Wikipedia articles. So, to expand the Brazilian Portuguese Wikipedia, we address the task of automatically generating the first section of a Wikipedia article, i.e., a *lead*. The lead section of a Wikipedia article is the first section, without a section title, that summarizes the article's content, as highlighted in Figure 1.

In particular, in this work, we propose an algorithm to automatically generate the lead section of Wikipedia articles from a set of online descriptions as a *multi-document abstractive summarization* (MDAS) of reference websites. In the *multi-document summarization* (MDS) approach, a single summary is written from multiple source documents. *Abstractive* summarizers generate the output by building new sentences, either by rephrasing or using words from the model's vocabulary, rather than simply extracting the crucial sentences from the input, as is done in extractive summarization (GUPTA; GUPTA, 2019).

Our survey on related work showed that MDAS is overlooked if compared to more "traditional" Natural Language Processing (NLP) tasks, such as Single Document Summarization (SDS) or Machine Translation (MT). The main difficulty in the field is related to the considerable size of the input of several documents, both in the number of source documents and length.

[1] Wikipedia numbers extracted from https://pt.wikipedia.org/wiki/Wikipedia, visited in 10/03/2023.

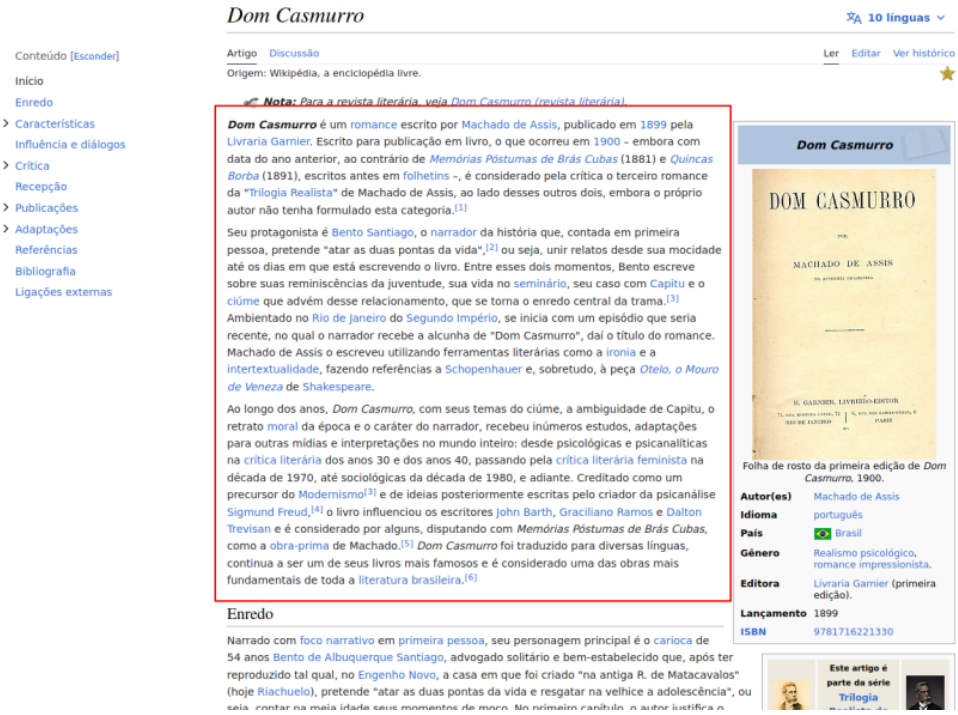


Figure 1 – Screen capture of the Brazilian Wikipedia page for Machado de Assis's book, "Dom Casmurro". The article's lead (highlighted) briefly summarizes the described concept. (source: Wikipedia, available at: https://pt.wikipedia.org/wiki/Dom_Casmurro, visited in 10/03/2023).

On the one hand, more information should support the writer's complete vision of the described subject. But having multiple documents to summarize imposes the challenge of fine-grained content selection, an arduous task for automatic summarizers. The algorithm leverages information from several sources and selects a few to compose the final summary without prior knowledge. Also, the considerable amount of text to process makes it difficult to interpret and store specific information from the input.

Despite this, recent works made some advances in the field. Authors were able to circumvent such difficulties and automatically create clear summaries by adapting state-of-the-art algorithms in NLP to the multi-document input scenario. In this context, one inspiring work for our investigation is WikiSum (LIU et al., 2018), as it also addresses the generation of the lead section of Wikipedia through the summarization of websites. Similarly to it, we apply a framework with extractive and abstractive stages. We reproduce some of the authors' experiments and further test new techniques for both stages. In addition, our focus is on summarizing texts in Portuguese instead of English.

Our final goal is to create a service that generates Brazilian Wikipedia articles by automatically searching and summarizing available documents on the web. This tool would help spread and democratize knowledge, as people could look up up-to-date descriptions

on virtually any unprecedented topic. Also, the generated descriptions could be used in a practical application as a template for Wikipedia, reducing the moderator's work to only checking the reports before they are released. This way, we could improve the coverage of Wikipedia in Brazil and elsewhere.

1.1 Objective

We propose in this work the PLSum, the *Portuguese Long Summarizer*, an MDAS framework specialized in the generation of wiki-like Portuguese descriptions for multiple documents, in which the abstractive stage is a sequence-to-sequence (*seq2seq*) Transformer neural network (VASWANI et al., 2017).

Seq2seq models transform an input sequence of tokens (i.e., words, subwords, punctuation, numbers) into another sequence of tokens. By doing so, this type of architecture can perform abstractive summarization. On the other hand, transformers are the state-of-the-art neural network architecture for most *seq2seq* tasks, such as automatic summarization, machine translation, and question answering (BAHDANAU; CHO; BENGIO, 2015; DEVLIN et al., 2018; RAFFEL et al., 2020).

For the extractive stage, we explore using dense semantic representations of sentences to perform a fine-grained content selection. More specifically, we apply clustering of sentence embeddings extracted from pre-trained Transformers.

In particular, we highlight two specific objectives:

- Compare state-of-the-art Transformers architectures for the MDAS task in Brazilian Portuguese. In particular, we assess the effectiveness of *pre-trained seq2seq Transformer neural networks*. The concept behind pre-training is to train a model on unsupervised language modeling tasks to learn contextual representations of sentences that can be further specialized with fine-tuning on specific tasks. Indeed, pre-trained Transformers are the state-of-the-art for numerous *seq2seq* tasks (DEVLIN et al., 2018; RAFFEL et al., 2020). Notably, we fine-tune, and test PTT5 (CARMO et al., 2020), which is pre-trained on Portuguese data.

- Perform a comparison between *sparse and dense strategies* to perform content extraction from input texts in the extractive stage. Sparse strategies are the ones that use word matching between the query and the source text to filter relevant input extracts. On the other hand, dense strategies compare continuous vector representa-

tions, *i.e.*, embeddings, to retrieve relevant information. While sparse strategies are more straightforward than dense, recent studies indicate that the second is superior in retrieving semantic correlations between query and source text from numerous sources (KARPUKHIN et al., 2020).

1.2 Accomplishments

Throughout this research, we approached the subject of multi-document summarization from several angles of analysis. Here, we gather the principal analysis and results achieved, some of which were recognized by other researchers and institutions in the community. We highlight the following accomplishments.

In 2020, we published an investigation about sentiment-aware extractive summarization of reviews (OLIVEIRA; COSTA; HRUSCHKA, 2020). Later, we published a homonyms paper to this dissertation at *Encontro Nacional de Inteligência Artificial e Computacional* (ENIAC) 2021 (OLIVEIRA; COSTA, 2021). Here, we analyzed strategies to generate abstractive wiki-like summaries in Portuguese. Our results indicate that, when appropriately trained, *seq2seq* Transformers with sparse content selection can generate comprehensive abstractive summaries from reference websites in Brazilian Portuguese. We also presented the *BRWac2Wiki* dataset in the same work, a dataset generated from thousands of pairs ⟨ websites, wiki summary ⟩. This article won second place in the ENIAC 2021 BEST MAIN TRACK PAPERS award.

Practically, we contributed not only with the extensive analysis of the topic but also with:

- A summarization dataset in Portuguese with more than 100000 training and validation samples, BrWac2Wiki (https://github.com/aseidelo/BrWac2Wiki).
- Code to reproduce the analysis and apply the full framework, with different options of internal algorithms (https://github.com/aseidelo/wiki_generator and https://github.com/aseidelo/clustersum).
- A demonstration of such a summarization tool, where users can load reference documents and automatically generate a Brazilian Portuguese wiki-like summary (https://huggingface.co/spaces/seidel/plsum_autowiki).

It is also worth noting that our studies on Brazilian Portuguese NLP also resulted in

relevant cooperation in related fields, such as in question answering (CAÇÃO et al., 2021; PASCHOAL et al., 2021) and zero-shot text classification (ALCOFORADO et al., 2022).

1.3 Organization of manuscript

This manuscript is organized as follows: First, we explain fundamental concepts associated with automatic summarization and *seq2seq* generation in Chapter 2. Then, we present state-of-the-art publications on multi-document summarization in Chapter 3.

In Chapter 4, we show *(i)* an overview of the architecture for PLSum, the Portuguese multi-document abstractive summarizer; *(ii)* the implementation of the sparse extractive step and dense abstractive step of the framework; *(iii)* the dataset created to train, validate and test the model, BRWac2Wiki; and *(iv)* tests and results.

In Chapter 5, we take a closer look at the extractive stage, and analyze the viability of exploring dense semantic representations to improve content selection. By doing so, we address some problems of sparse content selection strategies, such as the problem of redundancy in multiple source texts about the same topic. We introduce the chapter showing the theoretical advantages of dense semantic representations and how they could solve issues evidenced in the sparse extractive stage. Then, we propose a method based on clustering sentence embeddings to perform the extractive stage of our summarization framework. Finally, we show the final results and comparisons considering all the scenarios above (i.e., combinations of different extractive and abstractive methods).

In Chapter 6, we outline this research's final results and discuss its implications. We also share our opinion about possible advantages or disadvantages of the explored techniques for different use case scenarios. Finally, we present our thoughts on promising future studies on the topic, considering the international and Brazilian Portuguese state-of-the-art.

Ultimately, in Appendix A, we make a brief discussion about the big star of the moment in the NLP community, ChatGPT. We also compare summaries generated by ChatGPT and our model.

2 BASIC CONCEPTS

This chapter introduces the main concepts and techniques associated with MDAS that are common or applied in this work. Firstly, we introduce the field of automatic summarization. Then, we discuss *seq2seq* supervised learning strategies, notably Transformers neural networks, the state-of-the-art for several *seq2seq* tasks. Finally, we describe common strategies to evaluate the quality of the summaries.

2.1 Automatic Summarization

Summarization is defined by Jones et al. (1999) as a *"reductive transformation of a source text to summary text through content condensation by selection and/or generalization on what is important in the source"*.

As described before, automatic summarization models might be either extractive, when it chooses prominent sentences from the input to compose the summary, or abstractive, when it makes an authorial summary with unprecedented sentences. In this work, we focus on abstractive summarization. Generally speaking, abstractive summarization strategies need to accomplish three tasks (LIN; NG, 2019):

1. ***Information extraction:*** extracts a convenient representation of the information contained in the input text (*e.g.*, via generating word or semantic graphs, latent vector representations, semantic role labeling).

2. ***Content selection:*** selects a subset of the extracted information to compose the summary (*e.g.*, via data clustering and aggregation, and graph pathfinding).

3. ***Surface realization :*** generates a summary based on the selected content, taking into account grammatical and syntactical rules of the output language (*e.g.*, via sentence compression and merging, templates, or sequential word choosing).

We illustrate the referred perspectives of MDAS in Figure 2.

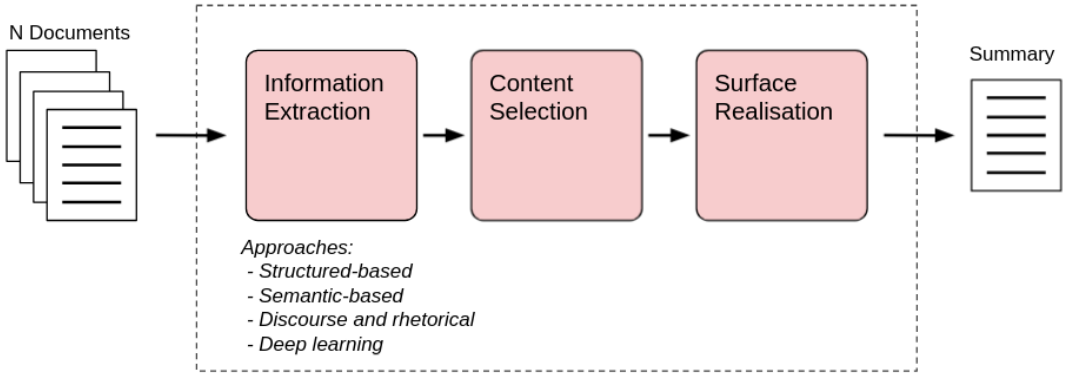


Figure 2 – Multi-document abstractive summarization general steps and approaches.

For instance, on Bing et al. (2015) information extraction is performed by labeling sentences with the *Predicate-Argument Structure* (PAS) (MARCUS et al., 1994) (i.e., by its syntactic role). Then, they perform content selection through a position-based salience score of n-grams in the extracted sentences, where n-grams are consecutive sequences of terms in the sentence. Finally, they perform surface realization by merging extracts from different sentences with an objective function based on salience and similarity scores

PAS is a syntactic tree representation that divides sentences into hierarchical syntactic categories, such as *Noun Phrases* (NP), and *Verb Phrases* (VP). NPs are self-contained extracts where the fundamental element is a noun or refers to a noun, while VP's fundamental element is the main verb, accompanied by an argument. There might be multiple levels of categories in subdivisions of the same extract. This classification results in a *constituency tree* of the sentence, with several levels of NPs and VPs, and single words as leaves. Figure 3 shows an example of a PAS constituency tree with additional *Part-of-Speech* (POS) tagging next to the words for the phrase *Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.*

Gupta and Gupta (2019) classify abstractive summarization strategies taking into account the approach employed to perform the task:

1. ***Structure-based approach***: explores pre-defined data structures — such as graphs, trees, and templates — to create the summaries.

2. ***Semantic-based approach***: explores semantic representations of input documents, i.e., semantic role labeling, PAS.

3. ***Discourse and rhetorical structure approach***: captures discourse relations among statements from different documents, such as equivalences, contradictions, or elaboration.

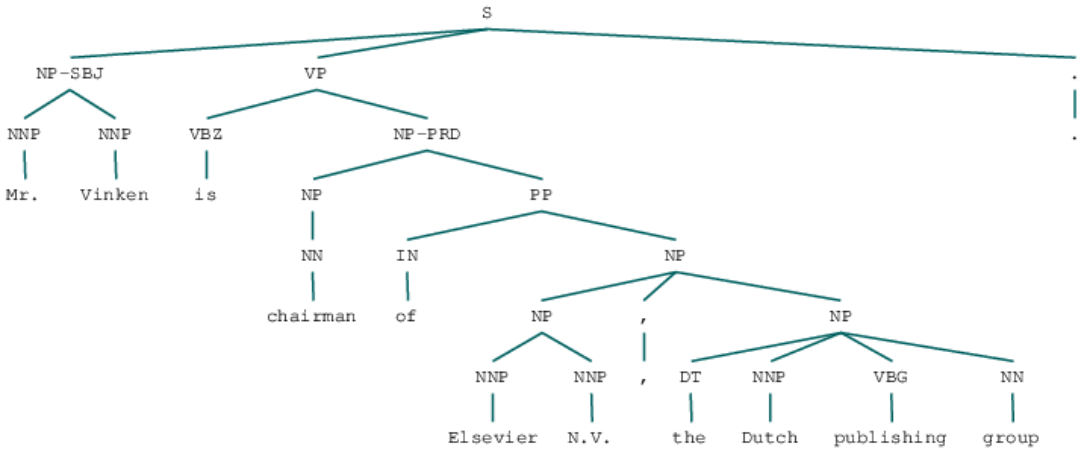


Figure 3 – Syntactic tree example annotated with PAS and POS tagging. Annotations on the tree leaves display POS information, as well as verb tenses, according to PropBank guidelines (source: extracted from PropBank (KINGSBURY; PALMER, 2002)).

4. ***Deep learning approach***: explores statistical learning methods, such as neural networks or reinforcement learning, to perform the task.

Regarding multi-document input, the additional challenge of extracting content from different sources implies that the model will have to deal with more redundant and unimportant information (LIN; NG, 2019).

To address this problem, authors either apply extractive summarization methods to reduce the input size before the information extraction stage or leave it to the content selection stage after the information extraction on all entry documents. For instance, Liu et al. (2018) and Lebanoff, Song and Liu (2018) apply extractive summarization techniques on the concatenated source documents before applying their neural models. On the other hand, Coavoux, Elsahar and Gallé (2019) cluster *document embeddings* after the neural model processes them. Documents embeddings are continuous vector representations of documents that regard the semantic information of the sentences.

There are several techniques in the literature to generate abstractive summaries. Early works on automatic summarization mostly applied template-based strategies, e.g., by filling pre-defined templates with crucial information from the input (GENEST; LAPALME, 2011; GERANI et al., 2014), or sentence compression and merging strategies (BING et al., 2015). Later, supervised abstractive techniques prevailed, majorly because of the success of *seq2seq* neural networks approaches, such as encoder-decoder Recurrent Neural Networks (RNNs) and Transformers.

2.2 *Seq2seq* Neural Networks

Seq2seq neural networks receive a sequence of known tokens from a fixed vocabulary and output another sequence of tokens. As already mentioned, tokens are the pre-defined building blocks of a sentence. For instance, tokens might be words, numbers, syllables, or characters.

We formalize the abstractive *seq2seq* task as follows: given (1) a fixed vocabulary V with M known tokens

$$V = \{t_1, t_2, ..., t_M\}, \tag{2.1}$$

where M is the size of the vocabulary, and (2) an input sequence $\mathbf{x}$ with J tokens,

$$\mathbf{x} = (x_1, x_2, ..., x_J),\ x_i \in V \quad \forall i \in \mathbb{N} : [1, J], \tag{2.2}$$

where x_i is a random variable of possible tokens. The objective is to generate a sequence $\hat{\mathbf{y}}$ of size K,

$$\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_K),\ \hat{y}_i \in V \quad \forall i \in \mathbb{N} : [1, K], \tag{2.3}$$

that maximises the theoretical distribution $p(\mathbf{y}|\mathbf{x})$, the probability of having the best $\mathbf{y}$ given $\mathbf{x}$, for all possible combinations of $\mathbf{x}$ and $\mathbf{y}$,

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}). \tag{2.4}$$

By considering the conditional distribution independent and identically distributed (i.i.d.) and applying the chain rule, we have:

$$p(\mathbf{y}|\mathbf{x}) = p(y_1, y_2, ..., y_K|\mathbf{x}), \tag{2.5}$$

$$p(\mathbf{y}|\mathbf{x}) = \prod_{k=1}^{K} p(y_k|y_{k-1}, ..., y_1, \mathbf{x}). \tag{2.6}$$

So, the distribution for every kth token is computed given the last output tokens $(y_{k-1}, ..., y_1)$ and the input $(\mathbf{x})$. For simplicity, authors generally compute the negative log-likelihood, that is:

$$-log(p(\mathbf{y}|\mathbf{x})) = -\sum_{k=1}^{K} log(p(y_k|y_{k-1}, ..., y_1, \mathbf{x})). \tag{2.7}$$

Seq2seq models try to estimate $p(y_k|y_{k-1}, ..., y_1, \mathbf{x})$ as a function of its trainable parameters $\boldsymbol{\theta}$. We refer to this estimation as $\bar{p}_{\boldsymbol{\theta}}(y_k|y_{k-1}, ..., y_1, \mathbf{x})$.

Supervised models learn the set $\boldsymbol{\theta}$ by observing examples on large datasets. For instance, let us say we have the dataset $(\mathbf{X}', \mathbf{Y}')$ where $\mathbf{X}' = (\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^N)$ is the input sequence, $\mathbf{Y}' = (\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^N)$ the output sequence, previously labeled (e.g., by specialists), and N is the number of examples of the dataset. One strategy to infer $\bar{p}_{\boldsymbol{\theta}}(y_k|y_{k-1}, ..., y_1, \mathbf{x})$ is to apply the maximum likelihood estimation, where the elements of $\boldsymbol{\theta}$ are chosen to maximize the likelihood of $\mathbf{Y}'$ given $\mathbf{X}'$, that is $p(\mathbf{Y}'|\mathbf{X}')$ (and thus minimize the log-likelihood). By considering the examples on the dataset to be i.i.d., the likelihood and negative log likelihood of $\mathbf{Y}'$ given $\mathbf{X}'$ are:

$$p(\mathbf{Y}'|\mathbf{X}') = \prod_{i=1}^{N} \bar{p}_{\boldsymbol{\theta}}(\mathbf{y}^i|\mathbf{x}^i), \tag{2.8}$$

$$-log(p(\mathbf{Y}'|\mathbf{X}')) = \sum_{i=1}^{N} -log(\bar{p}_{\boldsymbol{\theta}}(\mathbf{y}^i|\mathbf{x}^i)), \tag{2.9}$$

$$-log(p(\mathbf{Y}'|\mathbf{X}')) = -\sum_{i=1}^{N}\sum_{k=1}^{K} log(\bar{p}_{\boldsymbol{\theta}}(y_k^i|y_{k-1}^i, ..., y_1^i, \mathbf{x}^i)). \tag{2.10}$$

So, the model fits $\boldsymbol{\theta}$ so that the sum of negative log predictions is minimal for the training set.

Having this estimation of conditional probabilities, the best inference $\hat{\mathbf{y}}$ given any $\mathbf{x}$ is the combination of tokens with the smallest sum of negative log-likelihoods. As it would be computationally inefficient to try every possible combination of tokens (M^K possibilities), the models also apply a searching strategy to infer $\hat{\mathbf{y}}$. The naive (and most common) strategy for performing the search is always to infer $\hat{y}_k$ as the token with the highest "local" conditional probability. However, this might not result in the globally optimal set.

2.3 Encoder-Decoder Networks

The standard neural network architecture for *seq2seq* generation is the *encoder-decoder*, which was brought roughly at the same time by Sutskever, Vinyals and Le (2014) and Cho et al. (2014) for the task of machine translation. This architecture has encoding and decoding steps: Firstly, the encoder layers process the input sequence $\mathbf{x}$ in such a

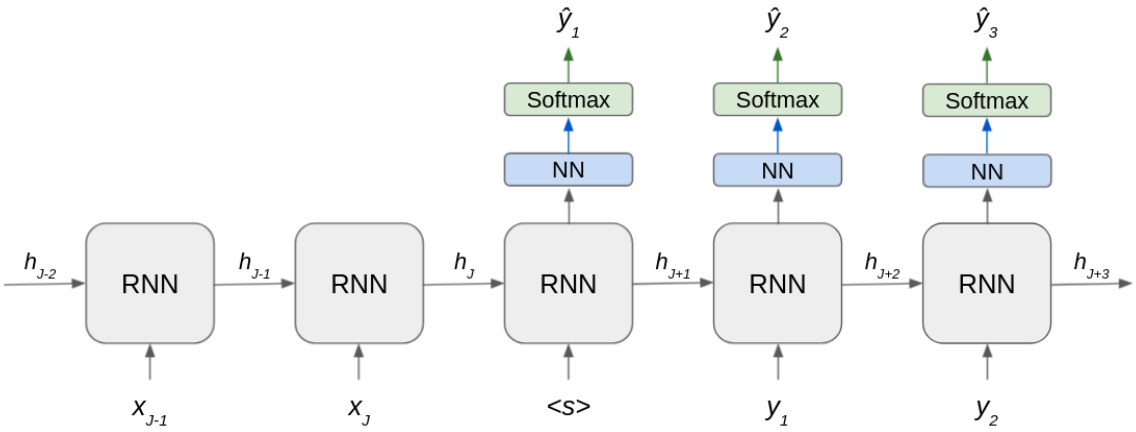


Figure 4 – RNN encoder-decoder: The decoder utilizes the last state of the encoder (h_J) as well as the embedding of a starting signal ($< s >$) to start generating outputs through a classification of tokens over softmax results. Posterior decoder layers use the last decoder state (h_{J+i}) to predict the next token.

way as to transform the sequence into a vector of continuous values that keep regard to semantic relationships among different sentences. Then, step by step the decoder infer $\hat{y}_k$ by estimating $\bar{p}_{\boldsymbol{\theta}}(\hat{y}_k|y_{k-1}, ..., y_1, \mathbf{x})$ and chooses the next token $\hat{y}_k$ as the one with the highest probability.

So, $\bar{p}_{\boldsymbol{\theta}}(\hat{y}_k|y_{k-1}, ..., y_1, \mathbf{x})$ is estimated as a non-linear function of the last input y_{k-1} and last decoder state h_{k-1}, which should carry information on the last states, including the ones from the encoder. Figure 4 illustrates the RNN *seq2seq* generation procedure.

To estimate the conditional probability of the next token, authors often use RNNs, whose neurons are recurrently fed with past outputs to permit the consideration of past states in present predictions (GRAVES, 2013). More specifically, both Cho et al. (2014) and Sutskever, Vinyals and Le (2014) utilized Long Short-Term Memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) as the non-linear function, a variation of RNN that applies gating mechanisms to control information flow through memory units on each time step. By doing so, it has fewer gradient vanishing and exploding problems in deep models than traditional RNNs. A lighter version of LSTM, likewise widely utilized, is the Gated Recurrent Units (GRUs) (CHO et al., 2014).

The authors also applied pre-trained word embeddings to reinforce the model's capability to capture the semantic meaning of words. Word embeddings are real-valued vector representations of words that keep semantic correlations in the multidimensional space (PENNINGTON; SOCHER; MANNING, 2014). They are pre-trained in context-capturing tasks, such as skip-gram or continuous-bag-of-words (CBOW) (MIKOLOV et

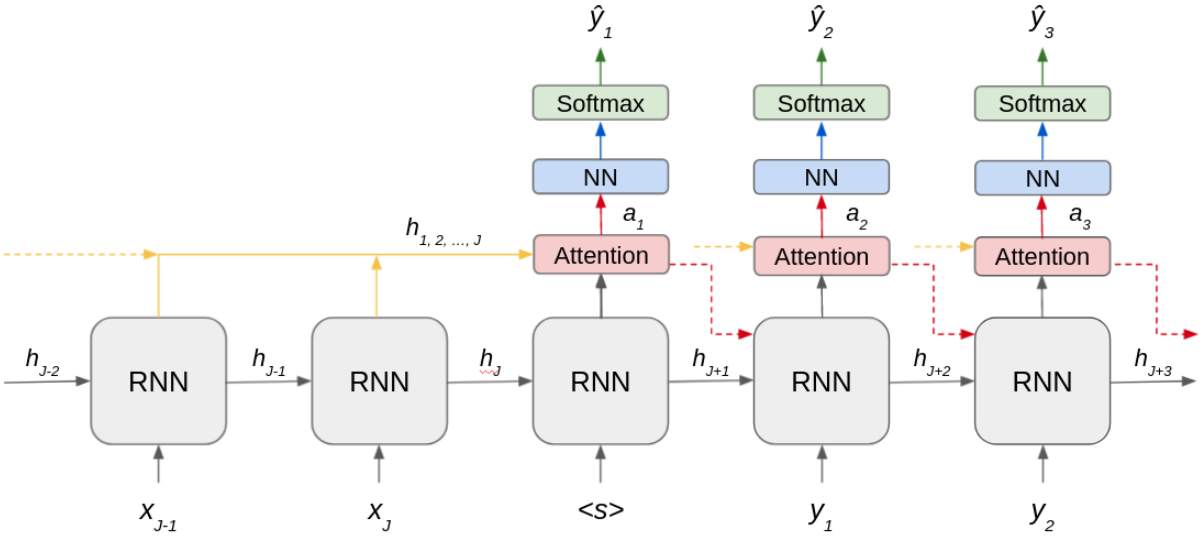


Figure 5 – Attention RNN encoder-decoder: Past states from the encoder ($h_1, h_2, ..., h_J$, in yellow) are stored and used to calculate the attention (a_i, in red) for each decoding step. The resulting vector is also applied to calculate the next decoder state.

al., 2013) on large corpora. Then, the embeddings are incorporated as input vector representations of tokens, permitting a meaningful semantic representation. Word2Vec (MIKOLOV et al., 2013) and GloVe (PENNINGTON; SOCHER; MANNING, 2014) are examples of pre-trained embeddings utilized in the MDAS literature.

Plain RNN encoder-decoders cannot correctly generate long sequences, as past tokens rapidly lose influence on subsequent tokens values. To deal with that problem, Bahdanau, Cho and Bengio (2015) created *attention*, a mechanism that allows the neural encoder-decoders to search for relevant parts in the source sequence at each step generation. It works by storing intermediate vector states at each encoding step and then learning to correlate each of these stored encoder states with the current decoder state at a given step of output generation. So, the decoder output at step k is now a non-linear function of every encoder state $h_1, ..., h_J$, the last decoder attention output a_{k-1} and y_{k-1}. Figure 5 illustrates an attention RNN encoder-decoder.

If the attention accounts for all encoder states for every decoding step, it is called *global attention*. In contrast, if only a limited window of encoder states is considered, it is called *local attention*. The RNN encoder-decoder with attention was vastly applied to abstractive summarization publications (RUSH; CHOPRA; WESTON, 2015; CHOPRA; AULI; RUSH, 2016; VASWANI et al., 2017).

2.4 Transformers Neural Networks

Inspired by the success of attention-based models, Transformer (VASWANI et al., 2017) was created to address the *seq2seq* task with multiple *self-attention heads* and no recurrent layers. In the next paragraphs, we will briefly describe Transformer's main features. We recommend refering to Vaswani's article for an extensive explanation of the model.

In this model, matrices (word embedding dimension on $columns \times sequence$ dimension on lines) represent input sequences. The model processes the input sequence simultaneously because there are no recurrent layers. As a result, the decoder masks future tokens (not yet generated) so that the decoder layers only see past predictions at each time step. The lack of recurrence also removes the ability of the model to understand the ordering of tokens, so a positional encoding is added to regular input embeddings to differentiate token positions on the sentence.

The architecture contains several blocks of self-attention layers (red blocks in Figure 6) concatenated with feed-forward networks (blue blocks). The attention layer output is defined as a function of Key (Key), query (Query), and value (Value) vectors of dimension dim. The model uses parallel "heads" for computing attention: the input Key, Query, and Value matrices are split and processed by independent attention "heads". Figure 6 illustrates the Transformer architecture.

In the original version of Transformer (VASWANI et al., 2017), the attention function is the "scaled dot product", defined as:

$$Attention(\text{Key, Query, Value}) = \frac{softmax(\text{Query} \cdot \text{Key}^T)}{\sqrt{dim}}\text{Value}. \tag{2.11}$$

Attention layers might be (1) encoder-to-decoder attention, with Query and Value from the last encoder layer and Key from the last decoder layer, or (2) encoder-to-encoder self-attention, i.e., Key, Query, and Value from the last encoder layer. The name self-attention means the three parameters are calculated from the same sequence.

This method is the absolute state-of-the-art for several *seq2seq* tasks (DEVLIN et al., 2018; RAFFEL et al., 2020). However, it has the drawback of being computationally expensive because it has to store several encoder and decoder states at each step.

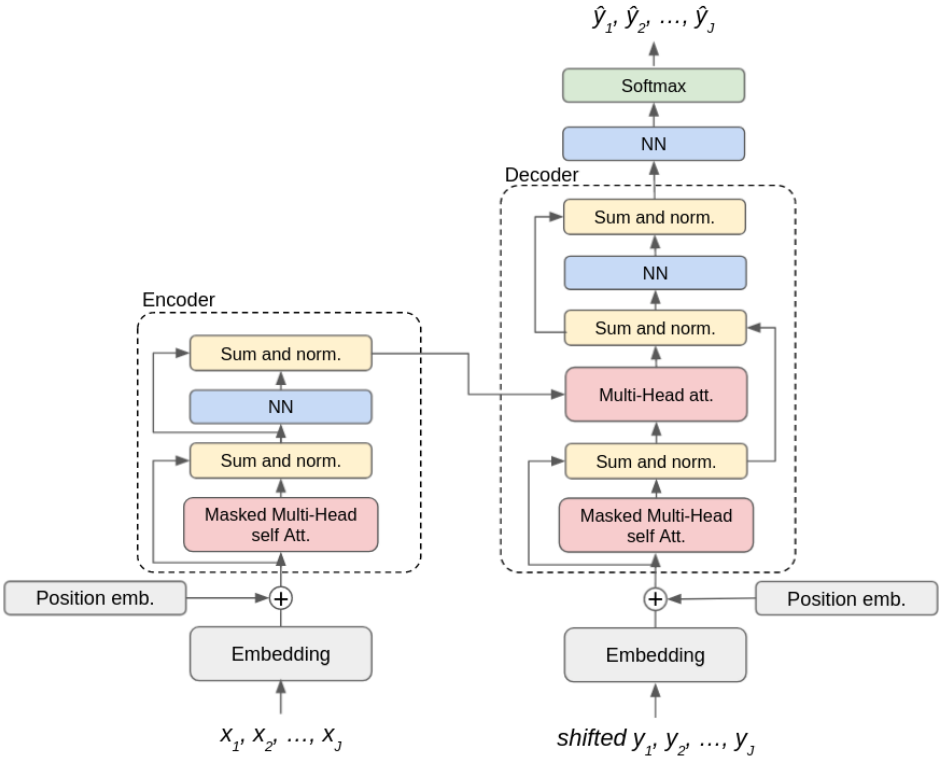


Figure 6 – Transformer encoder-decoder illustrated: x_i are input tokens, y_i the target sequence tokens, and $\hat{y}_i$ are predicted tokens at each time step. Position embeddings are added to input embeddings, and the resulting vector is processed by stacked layers of multi-head attention and feed-forward networks (NN). Decoder input is shifted and masked to only produce one output at each time step.

2.5 Evaluation of Summaries

Abstractive and extractive MDS models are typically evaluated with the same metrics as single-document summarizers, as they have the same output type. So, the methods described in this section are general summarization quality metrics eligible for single or multi-documents, and extractive or abstractive approaches. The evaluations can be either automatic or human-based, as described in the sequel.

Automatic measures are usually computed by comparing the generated summaries with human-written ones. So, this comparison requires standard datasets containing input (source documents) and output (summaries) pairs. Such automatic measures should process and evaluate the quality of many summaries while accounting for different mistakes.

Different automatic approaches tend to trade-off between precision coverage and simplicity. While complex metrics can be more precise and capture different mistakes, simple ones are more easily applied and understood. Therefore, researchers often resort to more

straightforward measures on standard datasets for comparison purposes, but they might also use complex metrics to highlight specific aspects of their methods.

BLEU (PAPINENI et al., 2002) is a precision score comparing n-grams of candidate summaries to n-grams of reference summaries. It measures the percentage of candidate n-grams from the summaries that match the reference's n-grams over the total amount of n-grams in the candidate summaries. The score generally uses 1 to 4 grams. Moreover, BLEU was initially applied for the evaluation of machine translation methods.

One drawback of BLEU is that it favors small candidate summaries, as it is a precision-based score. So, the ROUGE metric (LIN; OCH, 2004) was created, enabling the use of *recall* of n-grams matching. ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation and is a package of slightly different scores. These modalities are:

- ROUGE (N) is similar to the BLEU score. However, it considers the percentage of reference summaries n-grams that match the candidates over the total amount of n-grams in the reference summaries.
- ROUGE-L applies the F-measure calculation over two components based on the longest common subsequence (LCS) length between reference and candidate summaries. LCS accounts for subsequence matches with any number or size of gaps between elements, provided that they follow the same order.
- ROUGE-W is a weighted version of ROUGE-L, where LCS with consecutive matches is favored. It prioritizes candidate subsequences with smaller gaps than others, a feature that is impossible in ROUGE-L.
- ROUGE-S(N) accounts for the skip-bigrams matches between reference and candidate summaries. Skip-bigrams are any pair of words in their sentence order, in a gap of maximum size N.
- ROUGE-SU (N) is an extension of ROUGE-S that accounts for unigram matches in a gap of up to N and skip-bigrams. It alleviates the ordering constraint in ROUGE-S matches, as skip-bigrams are only accounted for when word pairs in the candidate summary are in the same order as the reference.

METEOR (DENKOWSKI; LAVIE, 2014) is another metric based on word comparison that features a synonym-matching pre-processing stage. They apply stemming on the texts, i.e., reducing words to their radical, and use a pre-defined synonyms table to

compare words. It calculates the precision and recall of unigram matches, and the final score is a weighted harmonic mean between these two values. The recall is weighted nine times more than the precision. A penalty proportional to the fewer possible chunks and inversely proportional to the number of word matches is also applied, to account for gaps and differences in word order. The authors argue that this metric correlates better to human evaluation than BLEU and ROUGE scores, but the calculations are more complex and depend on stemming and synonym tables.

Pyramid (NENKOVA; PASSONNEAU, 2004) is a score that captures the semantic correlation between candidate and reference summaries. It accounts for the amount of Summarization Content Units (SCUs) in a candidate summary that is also present in a corpus of multiple reference summaries. SCUs are defined as a set of clauses that express the same semantic content. A weight is assigned for each SCU, corresponding to the number of occurrences in the corpus of reference summaries. This way, more frequent SCUs will provide a higher Pyramid score to the candidate summary.

Human evaluations can also be applied to quantify the quality of a set of summaries. They have multiple-choice question-and-answer forms about candidate summaries. As they rely on specialist interpretation, they can capture nuances that automatic measures can not. Indeed, the aforementioned automatic measures positively correlate to human evaluations on standard datasets.

The downside is that human evaluations are harder to apply and, thus, less scalable. Another major drawback is that the results from different experiments are not comparable, even with the same script, as they are generated within different contexts. Many authors utilize this resource as self-contained evidence, usually submitting summaries from other methods to the same evaluation. For instance, one script for human evaluation is the set of questions created on the Document Understanding Conference (DUC)(OVER, 2004) from 2007[1].

We observe that ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4 are the most utilized automatic metrics for MDAS. Arguably, it happens because of the straightforward implementation of ROUGE metrics and good correlation to human evaluations, as shown by Lin and Och (2004). Also, the popularity of this score on summarization tasks is appealing for new publications, as it means a bigger comparison pool.

There are some implementations of rouge scores for R and Python languages, and the

[1] An example of the procedure for human evaluation is available at: https://duc.nist.gov/duc2007/quality-questions.txt.

major machine learning frameworks (i.e., PyTorch and TensorFlow) incorporate them as a standard validation metric.

As an example of an actual application of rouge scores, let us consider a situation where we want to compare two summarization algorithms. Also, we have a set of input texts and a target summary about football, written by specialists, that we trust as a good summary:

Target summary: "Football is a team sport where players use their legs, head, and torso to pass a ball and score goals with it."

The algorithms output the following predictions:

Predicted summary 1: "Football is a team sport where players use their legs and head to handle a ball and score goals with it."

Predicted summary 2: "Football is a sport where players score goals."

In Table 1, we compute the F-measure for ROUGE-1 (R1-F), ROUGE-2 (R2-F), and ROUGE-L (RL-F) for the predicted sentences in comparison to the target summary. As one can see, the predicted summary 1 is more similar to the target than the other. In fact, the first prediction is more informative.

Table 1 – Example of the F-measure ROUGE-1 (R1-F), ROUGE-2 (R2-F), and ROUGE-L (RL-F) scores for two samples. In this example, the predicted summary 1 is more similar to the target, according to all three ROUGE scores.

	R1-F (%)	**R2-F (%)**	**RL-F (%)**
Predicted summary 1	93.0	73.2	88.4
Predicted summary 2	53.3	35.7	53.3

3 RELATED WORK

In this chapter, we first present notable publications on MDAS from the last ten years. Then, we show multi-document summarization works focused on the Brazilian Portuguese language.

3.1 Multi-document Abstractive Summarization

Early works on MDAS were mostly semantic-based or structure-based models. As such, we start by presenting notable works associated with these strategies. While semantic or rhetorical graphs generally represented semantic-based models, word graphs and syntactic trees represented structure-based models. Both types of surface realization are generally performed by template-filling strategies or sentence compression and merging.

One possible semantic formalization for MDAS is the Information Item (InIt) (GENEST; LAPALME, 2011). InIts are defined as the smallest elements of coherent information in a sentence. They define entities and their relations, extracted with semantic role labeling strategies and PAS logic.

Other publications also proposed the utilization of word graphs for the MDAS task. Word graphs are directed graphs of adjacent words, where nodes are the words themselves, and the directed edges represent different subsequent word possibilities. For instance, in Banerjee, Mitra and Sugiyama (2015), sentences are firstly clustered with heuristic-driven similarity measures, and then word graphs are created for each cluster. Then, sentences for the summary are generated from path-finding between the start and end nodes on these graphs.

Structure-based and semantic-based models were mainly applied for MDAS in English before 2015. After that, deep learning approaches for sequence-to-sequence NLP achieved outstanding results in various tasks, such as Machine Translation, Question Answering,

and Summarization (BAHDANAU; CHO; BENGIO, 2015; RUSH; CHOPRA; WESTON, 2015; VASWANI et al., 2017).

Despite the success of neural models on the single document abstractive summarization task, adapting these techniques to multi-document input is not straightforward. Even models with attention cannot appropriately deal with long sequences and redundant information from multiple documents (LEBANOFF; SONG; LIU, 2018). In addition, before 2018, there was not a multi-document summarization dataset large enough for training such models.

PG-MMR (LEBANOFF; SONG; LIU, 2018) applied an LSTM attention encoder-decoder with Pointer-generator (PG) (SEE; LIU; MANNING, 2017) to address MDAS. In particular, PG-MMR combines PG with Maximal Marginal Relevance (MMR) (CARBONELL; GOLDSTEIN, 1998), an extractive technique to rank sentences. Pointer-generator is a technique that combines abstractive and extractive approaches by enabling the model to also copy words from the input documents directly to the summary via picking words associated with high probabilities from the context vector. PG works by estimating the probability distribution of the next token from an *extended vocabulary*, consisting of the model's vocabulary and out-of-vocabulary input document words. At each step, PG-MMR selects a subset of the most relevant sentences with MMR and then generates one summary sentence from this subset with the PG network. Steps are repeated until the summary size is reached.

Liu et al. (2018) proposed a decoder-only Transformer to generate Wikipedia's article leads from source documents. The authors utilized extractive summarization techniques, such as sentence extraction with Term Frequency-Inverse Document Frequency (TF-IDF) (RAMOS et al., 2003), and TextRank (MIHALCEA; TARAU, 2004), to rank paragraphs from source documents based on their similarity to the title, and then apply the decoder-only Transformer in a limited set of most relevant paragraphs. TF-IDF uses word frequency to assign its importance in a sentence compared to other sentences (see more in Section 4.1). On the other hand, TextRank defines weighted graphs of text units (nodes) and calculates the text unit's importance from the connections to others (edges). The decoder-only *seq2seq* model works as a language model, where at each step, the input and masked output are combined into a single sequence, and then the model infers the next token. They applied standard local attention and memory-compressed attention (T-DMCA) modules. The T-DMCA module utilizes stridden convolutional layers over the keys and values to reduce their size while keeping queries unchanged. This decoder-only variation of the transformer model with T-DMCA can process longer input sequences

with fewer parameters than general encoder-decoder transformers.

To address the lack of trainable data and dataset biases, Chu and Liu (2019) proposed MeanSum, an unsupervised MDAS framework with auto-encoders. Auto-encoders are models trained to output the input sequence to learn convenient internal vector representations utilized as embedding. So MeanSum consists of one auto-encoder module supposed to learn representations for each input review and one summarization module that learns to generate a summary similar to each of the reviews from their mean representation. Both modules utilize LSTM encoder-decoders, while encoders and decoders share weights.

Coavoux, Elsahar and Gallé (2019) proposed a similar auto-encoder model for input representations, but with the application of supervised aspect-based clustering (with fixed aspects) algorithm on the encoded representations of sentences to ensure coverage. Then, the algorithm aggregates clusters into a sole vector representation. Finally, the model concatenate decoded sequences for each aspect to generate the summary.

MatchSum (ZHONG et al., 2020) is a supervised extractive model that relies on siamese transformer neural networks to perform single and multi-document summarization. Their model is trained so that good candidate summaries embeddings are close to the concatenated source documents embedding. Random combinations of sentences from the input documents are combined to form candidate summaries.

Pasunuru et al. (2021) performed multi-document abstractive summarization with an encoder-decoder Transformer neural network, where a graph representation of input sentences are encoded in parallel to the plain texts. The decoder processes these encodings to generate the abstractive summary. Finnaly, Xiao et al. (2022) altered the masked language pre-training objective of a Transformer, so that the model accounts for aggregating information from different source texts.

3.2 Summarization in Brazilian Portuguese

We highlight the following works in the literature focused on Brazilian Portuguese summarization.

CSTSumm (JORGE, 2010) utilizes Cross-document Structure Theory (CST) annotations to perform extractive summarization on the news articles summarization dataset CSTNews (LEIXO et al., 2008). The CST is a specific set of semantic and discourse annotations for multiple documents. It identifies several types of relations between the

sentences of the document. For instance, these relations might be equivalences, contradictions, or generalizations. On CSTSumm, the authors utilized an input of multiple documents and their CST annotations and focused on methods to perform content selection and summary planning. They proposed several options for rule-based content selection operators over the CST representations.

Later, RSumm (RIBALDO et al., 2012) further investigated the use of graphs for relationship mapping between sentences and graph path-finding for summarization, establishing the current state-of-the-art for CSTNews. CSTSumm and RSumm have the drawback of needing the CST annotations to perform the summarization.

Silveira and Branco (2012) made a step towards generating abstractive summaries for multiple documents in Portuguese. Although this work does not propose a multi-document abstractive summarizer, it applies sentence compression techniques to enhance multi-document extractive summaries, thus creating unprecedented (i.e., abstractive) sentences. Their method work by identifying candidate nodes on a PAS constituency tree to remove sentence parts.

More recently, Paiola, Rosa and Papa (2022) fine-tuned a pre-trained Transformer in Portuguese language, PTT5 (RAFFEL et al., 2020), in several summarization datasets in Portuguese and other languages. It is a similar approach to what we did in Chapter 4. The main difference is that we explore different sentence extraction techniques and fine-tune our model in our own dataset, BrWac2Wiki, detailed in Section 4.3.

3.3 Datasets

Labeled summarization datasets relate several entry documents with a human-written summary, generally with a fixed number of input documents for each entry. In the case of multi-document summarization, datasets relate multiple related source documents to the ground-truth summaries.

Authors apply datasets to compare reference and automatically generated summaries, typically using standard automatic metrics, which we address in Section 2.5. Using standardized datasets and metrics, we can compare different model's empirical relatedness to the ground-truths. Another relevant use for summarization datasets is in training statistical learning methods. For instance, those mentioned above state-of-the-art deep learning models require a large amount of training input-output pairs.

In contrast, only a few standard datasets are available for MDS, and they are often

Table 2 – Document summarization datasets for single-document (SD) and multi-document (MD) summarization, its characteristics, and the number of input-output pairs.

Type	Dataset	Source	Summary	# pairs
SD	Gigaword (EN)	1st sentence of news article	8.3 words	4M
SD	CNN/Daily Mail (EN)	News article	56 words	312K
MD	DUC (EN)	10 related news articles	100 words	320
MD	TAC (EN)	10 related news articles	100 words	728
MD	CSTNews (PT)	3 - 5 related news articles	350 words	50
MD	WikiSum (EN)	10 - 15 documents	10-1000 words	2M
MD	Multi-News (EN)	2 - 10 documents	90-350 words	55K

Source: adapted from Lebanoff, Song and Liu (2018), Liu et al. (2018), Leixo et al. (2008), and Fabbri et al. (2019).

small, usually having only hundreds of document-summary pairs (LIN; NG, 2019). It happens due to the difficulty of creating hand-crafted ground truth for multiple related documents. Table 2 shows some well-known summarization datasets. We categorize them according to the application scenarios, *i.e.* for single-document (SD) or multi-document (MD) processing. All datasets are in English, except CSTNews, which is in Portuguese.

Some efforts towards the creation of an MDAS dataset were made at the Document Understanding Conference (DUC)(2000-2007) (OVER; DANG; HARMAN, 2007) and later at its successor, namely: the Text Analysis Conference (TAC) (2008-now) (DANG; OWCZARZAK et al., 2008). They compiled annotated MDS datasets with ten news articles associated with 100-word summaries, with 320 and 728 input/output pairs.

WikiSum dataset (LIU et al., 2018) was created in 2018. It consists of entries relating Wikipedia's articles leads to documents from the article's references and the most relevant web searches with the titles of the articles. By automatizing the data crawling, they generated a dataset with up to 2M examples. It has a variable quantity of reference documents associated with 10-1000 words Wikipedia leads. As far as we know, WikiSum is the first MDS dataset comparable in size to single-document summarization datasets, such as Gigaword and CNN/Daily Mail, and thus suitable for end-to-end neural training.

Later, Multi-News (FABBRI et al., 2019) was released. It consists of around 55.000 samples of multiple news about the same topic associated with a hand-written summary of these articles. Although Multi-News has considerably fewer samples than WikiSum, it can still be used to train and validate supervised models. This dataset also has the advantage of being more accessible and curated than WikiSum.

In the Brazilian Portuguese scenario, it is worth mentioning the MDS dataset CST-News (LEIXO et al., 2008), composed of 50 examples of summaries for related news

from different newspapers. The examples are annotated according to CST, a semantic relationship mapping between documents about the same subject.

3.4 Research Gaps

From our survey on *seq2seq* strategies, we observe that while Transformers architectures have reached unprecedented results on many different datasets, there is much to be done. Models still display unreliable information, specially in scenarios where they interpret and generate long texts, which is often the case of MDAS application.

As mentioned before, WikiSum (LIU et al., 2018) utilizes a Transformer for this task, but the article does not explore pre-trained models. Pre-trained models are trained on general language modeling tasks before being trained on the specific task. Literature indicates that these pre-trained models have great potential to enhance Transformer results further (DEVLIN et al., 2018; RAFFEL et al., 2020).

In addition, we observe that most works that apply Transformers do not explore ways to benefit from their dense representations of sentences to perform the extractive stage. Most works focused on the abstractive stage while applying traditional sparse strategies for content selection. So, we recognize four major research gaps:

- MDAS in Brazilian Portuguese;
- Utilization of *seq2seq* Transformers for MDAS;
- Utilization of pre-trained models for general *seq2seq* in Brazilian Portuguese;
- Utilization of dense semantic representations to perform the extractive sentence selection.

To explore the gaps, we investigate the pre-trained Transformer models for Brazilian Portuguese, such as Bertimbau and PTT5 (SOUZA; NOGUEIRA; LOTUFO, 2020; CARMO et al., 2020). They use the same architectures as the original pre-trained Transformers, which have proven effective in datasets with long sequences.

This dissertation explores all four gaps with PLSum, advancing even more in the state-of-the-art for Brazilian Portuguese and the international scenario.

4 PLSUM: PORTUGUESE LONG SUMMARIZER

In this dissertation, we contribute with PLSum, a two-step architecture that combines extractive and abstractive stages.

The framework has as inputs: (1) The desired title for the summary, $Title$, (2) A vocabulary of tokens, i.e., numbers, punctuation, words, sub-words, $V = \{t_1, ..., t_M\}$, and (3) A set of documents related to this title, $\mathbf{d} = \{Doc_1, ..., Doc_D\}$, where M is the size of the vocabulary, and D is the number of documents.

PLSum reads the set of documents $\mathbf{d}$ and returns a *wiki-like summary* about the title $Title$. The summary is a sequence of tokens from V.

The extractive stage starts by dividing documents into sentences ending with a period, $Doc_i = (Sent_{i1}, ..., Sent_{iN(i)})$, where $Sent_{ij} \in Doc_i$ is a sentence with about 100 words, $j \in [1 \dots N(i)]$ and $N(i)$ is the number of sentences in $Doc_i \in \mathbf{d}$, which may vary from document to document. The sentences from all documents are then grouped into a super document, $\mathbf{O} = (Sent_{11}, ..., Sent_{1N(1)}, ..., Sent_{D1}, ..., Sent_{DN(D)})$.

Then, the extractive stage selects L relevant sentences from the input set of documents $\mathbf{O}$. The framework is model-independent, and in our preliminary experiments, we apply TF-IDF (RAMOS et al., 2003), and TextRank (MIHALCEA; TARAU, 2004) for the calculation of sentence relevance (see Section 4.1 for definition) and selection. In the next chapter, we also explore the use of dense semantic representations, for that matter.

The L most relevant sentences are selected and concatenated with the title $Title$ in a single output sentence, $Sent_{ext}$, following descending order of relevance and with a separator token $[SEP]$ between them:

$$Sent_{ext} = Title \text{ [SEP] } Sent_1 \text{ [SEP] } ...Sent_i \text{ ... [SEP]} Sent_L. \tag{4.1}$$

The abstractive stage receives this sentence $Sent_{ext}$ and outputs the sequence $\hat{\mathbf{y}}_{summ}$.

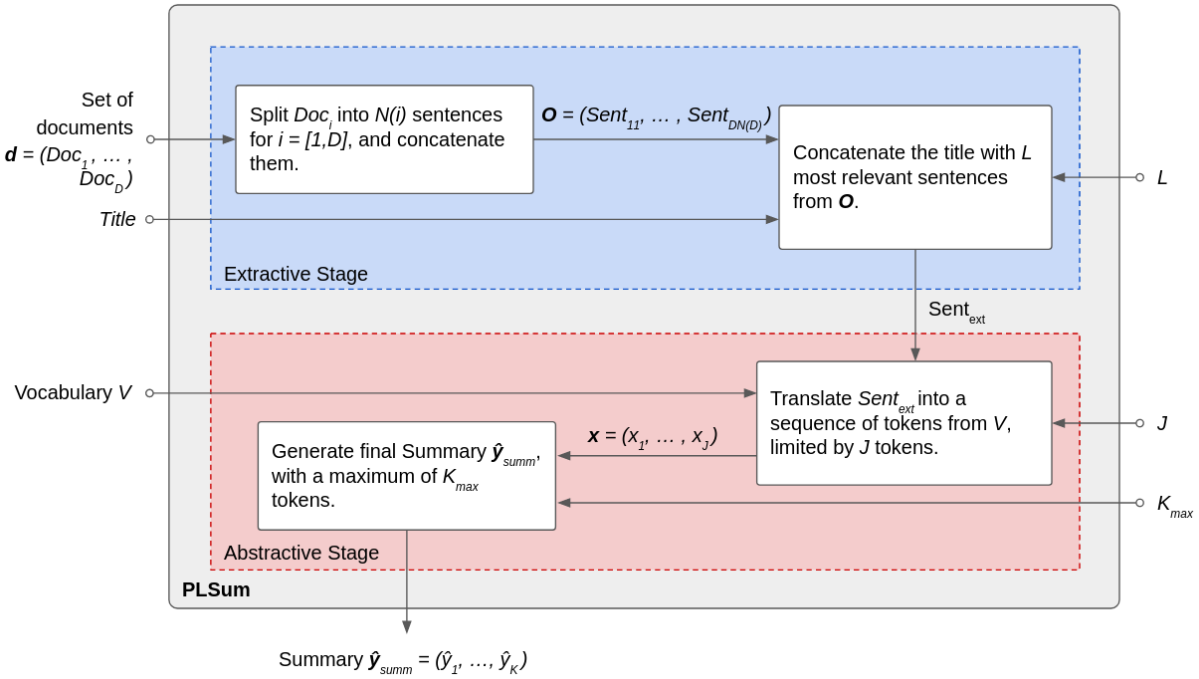


Figure 7 – PLSum flowchart. Given (1) the set of inputs: $Title$, vocabulary V, and a set of related documents $\mathbf{d}$ (left side parameters); (2) the set of hyperparameters: number of extracted sentences L, number of abstractive input tokens J, and the maximum number of output tokens K_{max} (right side parameters). PLSum filter relevant sentences and generate an authorial summary $\hat{\mathbf{y}}_{summ}$.

In particular, a summary is a sequence of tokens arbitrarily chosen from the vocabulary V (i.e., abstractive summarization). Thus it is not directly copied from the input, as in extractive summarization, as shown in Figure 7.

The abstractive stage starts by transforming numbers, punctuation, words, and sub-words from $Sent_{ext}$ into a sequence of known tokens from V, $\mathbf{x} = (x_1, ..., x_J)$. The sequence $\mathbf{x}$ is limited to J tokens, and this is a hyperparameter. Extracted sequences with less than J tokens are padded with a pre-defined padding token until they reach the size of J.

The abstractive summarization objective is to generate another sequence of tokens from V,

$$\hat{\mathbf{y}}_{summ} = (\hat{y}_1, ..., \hat{y}_K), \tag{4.2}$$

given $\mathbf{x}$. The number of tokens on the output sequence, K, is defined by the abstractive model (i.e., it may vary) and is limited by K_{max}, another hyperparameter.

4.1 Extractive Stage

The extractive stage extracts L relevant sentences from $\mathbf{O}$. Then, it outputs a single sentence, $Sent_{ext}$, composed of the summary title $Title$ and the L sentences separated by a pre-defined symbol, as shown in Eq. 4.1, where the pre-defined separator is "[SEP]".

We follow WikiSum (LIU et al., 2018) and test four candidates in the extractive stage, TF-IDF, TextRank, Random, and Cheating. TF-IDF and TextRank are well-known methods in the literature. *Random* and *Cheating* establish lower and upper bounds, respectively. The cheating score exclusively accounts for the target summary to generate an upper bound to the sparse extractive stage. It is supposed to answer how much better the extractive method can be and serve as an upper limit. Each extractive stage candidate is detailed below.

TF-IDF (RAMOS et al., 2003) scores relevance as a function of its relatedness to the user-defined title $Title$. TF-IDF stands for Term Frequency - Inverse Document Frequency and is a method for scoring a term's relevance in a sentence $Sent_{ij}$ in a sequence of sentences $\mathbf{O}$. The score is defined as:

$$TF - IDF_{term}(term, Sent_{ij}) = N_{term,Sent_{ij}}\ log(\frac{|\mathbf{O}|}{N_{sentences-term}}), \tag{4.3}$$

where $N_{term,Sent_{ij}}$ is the number of times the term appears in $Sent_{ij} \in \mathbf{O}$, and $N_{sentences-term}$ is the number of sentences in $\mathbf{O}$ citing the term at least once. TF-IDF assigns higher scores to frequent terms in $Sent_{ij}$, but not in the other sentences in $\mathbf{O}$. On the other hand, it assigns lower scores to terms that are either frequent or uncommon in all sentences in $\mathbf{O}$. Sentence relevance is defined as the sum of $TF - IDF_{term}(term, Sent_{ij})$ for every term in the title:

$$TF - IDF_{sentence}(Sent_{ij}) = \sum_{term}^{title\ terms} TF - IDF_{term}(term, Sent_{ij}). \tag{4.4}$$

Text-Rank (MIHALCEA; TARAU, 2004) is a graph-based sentence ranking model. A weighted graph is defined, where nodes V_i are chunks of texts (extracts of sentences), and weights w_{ij} are their relatedness to other chunks. The relatedness is infered by counting and normalizing co-occurrences of chunks in sentences. Then, the node's centrality is inferred as their centrality on the graph, similarly to how it is calculated on PageRank (PAGE et al., 1999),

$$TextRank(V_i) = (1 - d) + d * \sum_{V_j} \frac{w_{ij}}{\sum_{V_k} w_{jk}}, \tag{4.5}$$

where $d \in [0 \ldots 1]$ is a damping factor in charge of assigning a vertex weight of $(1-d)$ to unrelated chunks (chunks that never appear in the same sentence). Finally, the sentence rank is defined as the normalized sum of text chunks ranks for the sentence-level summarization.

Random is a technique that randomly chooses L sentences from $\mathbf{O}$. It indicates how the abstractive system behaves without an adequate extractive stage. This technique serves to define a lower limit for the extractive stage.

Cheating is a technique whose sentence score is calculated by recalling the *2-grams* between the sentences and the true *Wikipedia summary* for the given title. This summary is referred to here as the *target* summary:

$$Score_{Cheating} = \frac{\textit{2-grams}(Sent_{ij}) \cap \textit{2-grams}(target)}{\textit{2-grams}(target)}. \tag{4.6}$$

We approximate an theoretical upper bounds for the extractive stage with this formula. It is worth noting that this is a *sentence-level* "cheating" strategy, where the combination of the highest-score sentences (local optima) is considered the best summary. The actual best extractive summary given a metric can only be infered by calculating summarization scores for all possible sentence combinations, which is unfeasible. Such a strategy is classified as summary-level summarization.

4.2 Abstractive Stage

As mentioned, the abstractive stage starts by tokenizing $Sent_{ext}$, transforming it into $\mathbf{x}$. Then, it translates $\mathbf{x}$ into $\hat{\mathbf{y}}_{summ}$. To generate $\hat{\mathbf{y}}_{summ}$, we evaluate two supervised models, *PTT5* and *Longformer encoder-decoder*.

PTT5 (CARMO et al., 2020) is a *seq2seq* Transformer, based on T5 (RAFFEL et al., 2020), pre-trained on the Brazilian Portuguese corpus BrWac (WAGNER et al., 2018). The model architecture is similar to the original encoder-decoder Transformer (VASWANI et al., 2017), in which several blocks of self-attention layers and feed-forward networks are concatenated, as explained previously. PTT5 was pre-trained on BrWac for masked language modeling, where sentence tokens are replaced with a mask so that the model

has to guess them. We applied the "base" model[1], with 220M trainable parameters. The base version has 12 layers across the encoder and decoder and 12 attention heads.

Longformer encoder-decoder (BELTAGY; PETERS; COHAN, 2020) is another variation of the Transformer capable of processing longer inputs with the same model size as PTT5. Among other minor changes, Longformer exchanges global self-attention blocks (attention correlation across all tokens) by local attention, where self-attention is computed in a sliding window around the central token. Also, the dilated sliding window is applied, where the sliding window skips consecutive tokens to increase the receptive field without increasing computation. Despite the ability to have longer inputs, Longformer does not have a pre-trained checkpoint for Portuguese yet. Since there are reports of the effectiveness of transfer learning on seq2seq tasks (RAFFEL et al., 2020), the lack of pre-training is a drawback that might leverage the input size advantage.

We also applied the "base" version of Longformer encoder-decoder[2], with similar characteristics to PTT5: it has 6 layers on the encoder, 6 on the decoder, 12 attention heads on both, and sliding window attention of 256 across every layer. Also, we follow the recommendation from the authors and apply global attention only to the first token.

4.3 BRWac2Wiki: Training and Validation Dataset

To train the supervised models of the abstractive phase and validate the complete framework, we created the dataset BRWac2Wiki[3]. BRWac2Wiki is a structured dataset relating Brazilian websites to Wikipedia summaries. Each row has a *Wikipedia title*, the respective Wikipedia summary, and a list of documents related to the title.

The documents are website texts from BrWac corpus (WAGNER et al., 2018) that *cite every word from the title, in any order*. BrWac is a freely available Brazilian website corpus[4], composed of a list of records containing *url*, *title*, *text*, and *docid* (a unique id for each website). This criteria to relate documents and titles is vulnerable to ambiguous title words. However, it is up to the algorithm that will use the dataset to handle the content selection. Also, we applied post-processing, excluding rows on the dataset that did not regard the following rules:

[1] Available at ⟨https://huggingface.co/unicamp-dl/ptt5-base-portuguese-vocab⟩, visited in 10/03/2023.

[2] Available at ⟨https://huggingface.co/allenai/led-base-16384⟩, visited in 10/03/2023.

[3] The official repository for the dataset at the date of publication is ⟨https://github.com/aseidelo/BrWac2Wiki⟩.

[4] The homepage is at ⟨https://www.inf.ufrgs.br/pln/wiki/index.php?title=BrWaC⟩, visited in 10/03/2023.

1. A maximum of 15 documents for each summary;
2. A minimum of 1000 words in total on each set of documents;
3. A minimum of 20 words in each Wikipedia summary;
4. Examples were also subjected to clone detection, as defined in WikiSum:

$$P_{clone} = \frac{\textit{1-gram}(Doc_i) \cap \textit{1-gram}(a)}{\textit{1-gram}(a)}, \tag{4.7}$$

where websites Doc_i were compared to Wikipedia's summary a. Sentences with $P_{clone} > 0.5$ were excluded.

Table 3 details the dataset characteristics in percentiles. The "Input size" field shows the number of words of the concatenated input websites, the "Output size" is the analogous quantity for Wikipedia summary, and "N. documents" is the number of websites per example. One should interpret values associated with percentiles as the maximum values for the smaller x% of every feature. For instance, BRWac2Wiki output sizes range from 30 to 3846 words, while the input size range from around 8033 to over 1M words. Also, 80% of the outputs have less than 168 words.

Table 3 – Characteristics of BRWac2Wiki dataset in percentiles. Input and output sizes are in number of words.

Percentile (%)	20	40	60	80	100
Input size	8033	24210	53424	114777	1268802
Output size	30	49	86	168	3846
N. documents	2	8	15	15	15

Table 4 compares BRWac2Wiki and other summarization datasets for both English and Portuguese. For each dataset, column "# ex." shows the number of examples, "# docs/summ." shows the maximum number of input documents in an example, and "Task" briefly describes the challenge. Although it has 20 times fewer examples than WikiSum (LIU et al., 2018), BRWac2Wiki is 2000 times larger than the other Portuguese MDAS dataset, CSTNews (LEIXO et al., 2008), and is comparable in size to CNN/Daily Mail (HERMANN et al., 2015) single-document summarization dataset.

4.4 Preliminary Experiments and Results

Initially, in order to define the parameters of the extractive and abstractive stages, BRWac2Wiki was divided into train, validation, and test sets with 96782 (81%), 10753

Table 4 – Comparisson between BRWac2Wiki and another MDAS dataset. The column # ex. shows the number of examples for each dataset, # docs/summ. is the maximum number of documents per summary for each dataset, and Task is a concise description of the summarization task.

	Task	**# ex.**	**# docs/summ.**
CNN/Daily Mail (EN)	Gen. news highlight	312K	1
WikiSum (EN)	Gen. EN wiki	2M	over 1K
CSTNews (PT)	Gen. news summaries	50	3
BRWac2Wiki (PT) (ours)	Gen. PT wiki	119K	15

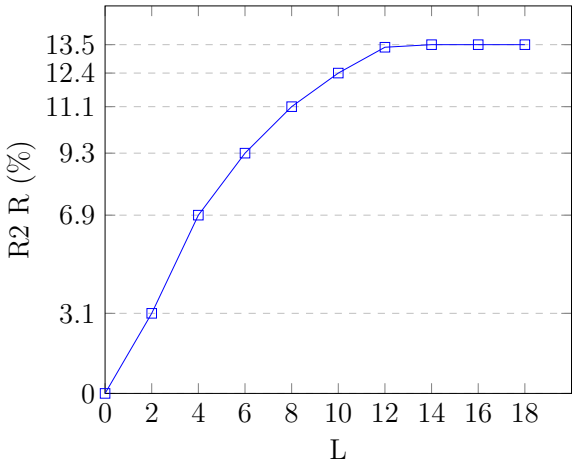


Figure 8 – ROUGE 2 recall of TF-IDF extractive stage on the validation set for different values of L (number of extracted sentences).

(9%), and 11948 (10%), respectively.

To define the desired input size for the abstractive models, we evaluated the recall of 2-grams (R2 R) between the extractive TF-IDF output and target on the validation set for different values of L (number of selected sentences), from 0 to 20. Figure 8 shows the results. One can notice that for $L > 12$, the R2 R measure does not increase over 13.5%, which indicates that further increasing L will not aggregate information for the abstractive model input. Thus, we define our desired value for L as 12; hence J should be around 1200.

We trained the abstractive candidates on an Intel Xeon CPU @ 2.30GHz with Tesla V100 GPU.[5] The limiting factor for input size was the GPU volume, so for PTT5, we could apply a maximum of $J = 768$, while for Longformer, we managed to apply $J = 1024$. In addition, we utilized the same output size $K_{max} = 256$ for both models, as 90% of the target summaries on the BrWac2Wiki dataset have less than 256 tokens, while the minimum size allowed was 20.

[5]Google Colab Pro specs on testing.

Both abstractive models (PTT5 and Longformer) were trained with similar hyperparameters: Beam search with the number of beams equal to 2, batch size of 16, and gradient accumulation steps equal to 2.

Then, we performed three experiments to validate our preliminary version of PLSum, that is, different versions of the extractive stage (content selection) and *seq2seq* Transformers abstractive stage:

Experiment 1: It is an ablation study of the candidates for the extractive stage. We apply *TextRank*, *TF-IDF*, *Random*, and *Cheating* versions of the extractive stage on the BRWac2Wiki test set. As 90% of the target summaries in BrWac2Wiki have less than 256 tokens, we applied $L = 5$ when utilizing the extractive stage only, thus $L < 10$. Then, we compute ROUGE scores and compare results. The difference between *TF-IDF* or *TextRank* and *Cheating* is an indication of how better the extractive stage could perform, while *Random* is a random baseline.

Experiment 2: We fine-tune and compare four pipelines of PLSum, also with ROUGE automatic evaluation on the BRWac2Wiki test set. We apply the best performing extractive model (*TF-IDF*) and compare it with *Random* to access the extractive stage relevance on the complete framework.

The compared pipelines are:

- **TF-IDF + LF (Longformer), J = 1024:** The model with the bigger receptive field, but without pre-train and local attention;
- **TF-IDF + PTT5, J = 768:** The maximum receptive field possible for PTT5, pre-trained on Portuguese and with global attention;
- **TF-IDF + PTT5, J = 512:** This is analogous to the previous one but with a smaller receptive field;
- **Random + PTT5, J = 768:** We apply Random as the extractive stage with PTT5 as the abstractive stage to assess the influence of the TF-IDF extractive in the complete framework.

Experiment 3: We apply the best-performing version of PLSum (on Experiment 2) to the CSTNews dataset. We compare PLSum ROUGE scores to the results reported in RSumm (RIBALDO et al., 2012) and CSTSumm (LEIXO et al., 2008), two extractive summarization works that were also applied to CSTNews. Our main objective is to assess the model's effectiveness on different datasets without specific

Table 5 – ROUGE scores for the extractive stage and different abstractive pipelines of PLSum. Extractive models with $L = 5$ (number of 100 symbol sentences to extract) and Abstractive with $K_{max} = 256$ (maximum number of tokens on output). We consider a confidence interval of 95% for boundary values within square brackets.

Type	Model	R1 F (%)	R2 F (%)	RL F (%)
Extractive	Random	16.3 [16.1, 16.5]	2.5 [2.5, 2.6]	9.1 [9.1, 9.2]
	TextRank	17.6 [17.5, 17.8]	2.4 [2.4, 2.5]	9.7 [9.6, 9.8]
	TFIDF	17.8 [17.7, 18.0]	3.6 [3.5, 3.6]	10.0 [9.9, 10.1]
	Cheating	**19.7** [19.5, 19.9]	**6.2** [6.1, 6.2]	**10.8** [10.7, 10.8]
Abstractive	TFIDF + LF, J=1024	19.7 [19.6, 19.8]	9.2 [9.0, 9.5]	19.8 [19.6, 20.0]
	TFIDF + PTT5, J=768	**32.0** [31.7, 32.4]	**14.9** [14.6, 15.2]	**25.6** [25.2, 25.9]
	TFIDF + PTT5, J=512	**32.0** [31.6, 32.3]	**15.0** [14.7, 15.3]	**25.6** [25.2, 26.0]
	Random + PTT5, J=768	29.2 [29.0, 29.6]	12.7 [12.5, 13.1]	23.5 [23.1, 23.8]

fine-tuning. We discuss the differences of PLSum and the advantages and disadvantages of our abstractive framework.

We report the minimum, mean, and maximum f-measure (F) of ROUGE 1, 2, L on test set samples. We applied a bootstrap re-sampling on the test set to estimate the boundary values (min., max.), generating ROUGE scores for 1000 random samples. Then, we considered the 2.5% and 97.5% percentiles of the ROUGE scores as the minimum and maximum boundaries, i.e., a confidence interval of 95%.

The results of **Experiment 1** are shown in the rows associated with the "Extractive" type in Table 5. The extractive methods have the expected performance for all ROUGE scores, that is, $R_{Random} < R_{TextRank} < R_{TF-IDF} < R_{Cheating}$. *TF-IDF* R1, R2, and RL F1 mean scores are 1.5, 1.1 and 0.9 above *Random*, and 1.9, 2.6 and 0.8 bellow *Cheating*, respectively. For *TextRank*, on the other hand, we observe that R1 and RL are 1.3 and 0.6 above random, while R2 had no significant difference to random (because of the overlapping ranges).

Thus, *TF-IDF* was better than *TextRank* for this dataset. While both models had some effectiveness as extractive methods for Brazilian Portuguese sentences, they could still be improved if compared to the *Cheating* method.

For **Experiment 2**, the "Abstractive" section of Table 5 compares the results for the different pipelines of the complete framework. The method with *TF-IDF + PTT5* had the best performance on every score. Also, every model with PTT5 achieved higher ROUGE scores than the one with Longformer. *Random + PTT5* has smaller ROUGE scores than *TF-IDF + PTT5*, as expected. Finally, altering the value of J in *TFIDF + PTT5* ($J = 512$ and $J = 768$) did not result in significant changes for this test set with

a 95% confidence interval.

Therefore, we reached the following conclusions from the first two experiments (see Table 5) :

1. TF-IDF and TextRank are effective as extractive summarization methods for most metrics, except TextRank R2, which was statistically equivalent to random;

2. The extractive stage is important, as TF-IDF only and TF-IDF + PTT5 had better results than *Random* only and *Random + PTT5*, with the same set of hyperparameters (10.0% and 25.5% against 9.1% and 23.5% RL F, respectively);

3. The extractive stage alone is not enough, as the full abstractive framework had considerably better results than the best possible extractive method (*Cheating*) (25.6% against 10.8% RL F);

4. PTT5 has considerably better results than Longformer (26.7% against 19.8% RL), probably due to the pre-training in Brazilian Portuguese data;

5. For PTT5, the exposure to a smaller receptive field ($J = 512$) displayed similar results to the version with a bigger receptive field ($J = 768$).

Table 6 is the result of Experiment 3, which compares PLSum, CSTSumm, and RSumm in the CSTNews dataset. One can notice that PLSum had worse results than the extractive techniques, especially for R2.

Table 6 – Scores of MDS models for Brazilian Portuguese and PLSum on CSTNews dataset.

Type	**Model**	**R1 F (%)**	**R2 F (%)**
Extractive	RSumm (RIBALDO et al., 2012)	41.9	**34.3**
	CSTSumm (LEIXO et al., 2008)	**38.6**	20.6
Abstractive	PLSum (ours)	28.6	8.8

However, when inspecting the resulting summaries and the characteristics of the CSTNews dataset, we noticed that:

1. As it was trained to mimic Wikipedia, most PLSum summaries described key entities (i.e., people, places, concepts) from the set of documents instead of a case report. For example, for the news set about the Brazilian swimming team winning the gold medal at the Pan American Games of 2007, with the title “Brazilian Swimming Team Win on Pan” (freely translated), PLSum generated the summary:

"Thiago Pereira (Rio de Janeiro) is a Brazilian Swimmer competing in Pan American Swimming Games." (freely translated), which is a description for one of the athletes mentioned on the news;

2. The CSTNews dataset has a high 1-gram recall when comparing the full entry to the target summaries (95.37 % against 77.87 % of BRWac2Wiki), which may imply an advantage for extractive summaries, given the relatedness of input and target.

So, we conclude from **Experiment 3** that the abstractive framework is unsuitable for generating summaries without specific fine-tuning. One should consider extractive summarizers if there is insufficient data for training or there is no need for surface realization on the summaries.

Finally, we categorized the strengths and issues in the abstractive summaries generated with the test set with a qualitative analysis. Table 7 shows examples and highlights the errors presented here.

On the strengths of PLSum, the model showed the ability to merge concepts from the input sentences without having been explicitly coded. For instance, the summaries for "Mario de Andrade" and "Santos Dumont" display birth and death dates between parenthesis (like Wikipedia) by compressing a sentence or merging information from multiple sentences. We can also emphasize that most abstractive summaries displayed information in the correct order. For example, biographical content often begins with the full name, followed by the date of birth and death, and then the events and accomplishments of the person described. We highlight that PLSum can read texts in multiple styles and write a wiki-like summary. This feature is a great advantage compared to sentence extraction techniques.

However, PLSum has some problems. Although most summaries have syntactic and semantic consistency, many display incorrect information related to quantities. Dates, geographic positions, population, and area of countries were found to be incorrect. From inspecting the automatically generated summaries, we argue that the abstractive model performs better when fed with the appropriate content. We noticed that when the extractive stage fails to retrieve enough information about the described title, it tends to "fill the gaps" with mocked data. For instance, in biographic summaries the model outputs random dates if the extractive stage fail to retrieve birth and death dates. Thus, this issue might be related to biases generated by missing information in training examples.

An essential factor in understanding the incorrectness in summaries is that BrWac2Wiki

Table 7 – Examples of automatically generated summaries (Predicted Summary) for some titles (Title). Wikipedia lead is displayed at Target Summary column. We highlight issues in yellow and annotate the type between parenthesis.

Title	Predicted summary	Target summary
Feudalismo	o feudalismo foi um modo de organização social e política baseado nas relações servo contratos servis. tem suas origens na decadência do império romano. o sistema feudal se caracterizou pela concessão de feudos, quase sempre em forma de terras e trabalho, em troca de proteção política e militar. esse contrato feito era selado por um juramento de fidelidade.	feudalismo compreende o sistema político , econômico e social que predominou na europa ocidental entre o início da idade média até a afirmação dos estados modernos , tendo seu apogeu entre os séculos xi e xiii . o conceito teórico foi criado nos séculos xvii e xviii (...)
Santos Dumont	alberto santos dumont (cabangu (WRONG), 20 de julho de 1873 são paulo, 23 de julho, 1932) foi um aviador brasileiro. foi o inventor do 14 bis, o primeiro avião a voar no brasil, e o primeiro piloto a voar na história do país.	alberto santos dumont (palmira , guarujá ,) foi um aeronauta , esportista e inventor brasileiro . santos dumont projetou , construiu e voou os primeiros balões dirigíveis com motor a gasolina . (...)
Mário de Andrade	mário raul de moraes andrade (são paulo, 9 de outubro de 1893 rio de janeiro, 10 de agosto (WRONG) de 1945) foi um poeta, ensaísta, crítico literário, poeta e ensaísta (REP) brasileiro.	mário raul de morais andrade (são paulo , 9 de outubro de 1893 são paulo , 25 de fevereiro de 1945) foi um poeta , romancista , musicólogo , historiador de arte , crítico e fotógrafo brasileiro . (...)
Coreia do Norte	a coreia do norte (,,) é uma república democrática (WRONG) da ásia, localizada no continente ásia e a maior cidade do país. sua capital é pyongyang, a moeda usada é a north korean won (kpw) e a população conhecida é de 21. 928. 228 habitantes. a população da coreia é de cerca de 2. 527. 000 habitantes, e sua população é de 3. 728, 3 milhões de habitantes s vezes. (REP, WRONG)	coreia do norte , oficialmente república popular democrática da coreia (rpdc em coreano: hanja: transl . chos n minjuju i inmin konghwaguk) , é um país no leste da ásia que constitui a parte norte da península coreana , com pyongyang como capital e maior cidade do país . (...)

has a redundant set of input texts since they were extracted from a crawling of the internet with simple filters. The dataset is noisy because there are (1) unrelated synonyms extracted with the title filter, and (2) the automatic crawler cannot filter website texts perfectly, so unusual symbols might appear.

On the other hand, the dataset is redundant because there are many "copycats" online, which result in several similar sentences from different documents. In this sense, although the TF-IDF extractive stage can somewhat avoid noisy sentences, we highlight two major problems:

1. It increases the problem of redundancy: As the algorithm scores similar sentences with similar grades, it tends to retrieve similar sentences from redundant datasets.

So, the coverage of the sentence selection step is compromised.

2. It does not interpret the semantics of sentences, which makes it unable to filter unrelated perfect homonyms. While some MDAS datasets are curated enough so that the input documents are always about the same topic, we argue that such control is not scalable for most practical applications. So, making robust algorithms in terms of content selection is desirable.

Thus, we propose the ClusterSum, described in the next chapter, aiming to reduce these two cited problems, which are: (i) reduce the redundancy of information in the sentences produced in the extractive process and (ii) reduce the impact of extracting phrases selected because they are subjects with identical spellings but not related to the desired title.

5 CLUSTERSUM: A CLOSER LOOK TO THE EXTRACTIVE STAGE

The discussion in the last chapter indicates that the TF-IDF extractive stage could handle sentence redundancy and sentences related to perfect homonyms [1] better. Not only TF-IDF but most sparse content selection techniques cannot interpret the semantics of sentences, which makes it hard to address the coverage of the extractive stage.

So, in this chapter, we propose a method to improve the extractive stage of PLSum. In particular, we intend to apply clustering techniques on sentence embeddings to define the central topics in input documents. That way, we can set the algorithm to retrieve at least one prominent sentence about each central topic, thus ensuring coverage and non-redundancy.

5.1 Exploring Semantic Representations

As shown in Chapter 2, some works use convenient structures such as graphs or pre-defined labels to generate semantic representations of sentences. Here, we use dense semantic embeddings generated by pre-trained neural encoders since recent studies show better results in text interpretation tasks, such as Semantic Textual Similarity (STS) (DEVLIN et al., 2018).

The basic concept behind deep semantic representations is to generate continuous vector representations of sentences, the embeddings, to spread sentence embeddings in the multidimensional space. So, different sentences should be distant from each other in the embedding space, while similar ones are close. Pre-training neural encoders achieve this on several natural language comprehension datasets, where samples of sentences with different types of relations are exposed to the model (i.e., entailments, contradictions, equality, divergence). By applying optimization criteria, one can achieve a set of parame-

[1] Perfect homonyms: is when words have the same spelling and pronunciation but with different meanings.

ters for the network that minimizes an optimization function, such as the cosine distance between two embeddings, so that sentence embeddings are spread conveniently in the multidimensional space. Then, these embeddings can be utilized as a dense semantic representation for specific applications.

It is essential to notice here that the distance between embeddings calculated via a distance metric (i.e., cosine distance) will only have a clear interpretation if the encoder is specifically trained for that (i.e., trained for reducing the distance for sentences labeled as similar). Otherwise, one might use cross-encoders: passing two sentences to an encoder at once so that it can compute their similarity as a regression task. However, comparing a large set of sentences pairwise is computationally expensive, which makes it unfeasible for a large set of input documents, such as in the MDAS task. Thus, we focus on the strategy of applying pre-trained encoders to generate embeddings that are comparable in their distances.

For that matter, Sentence-BERT (SBERT) (REIMERS; GUREVYCH, 2019) is a model pre-trained explicitly so that the cosine distance of embeddings is correlated to the similarity of sentences. It firstly applies a Transformer encoder that generates a matrix of dimensions $[N - tokens, N - dim]$, where $N - tokens$ is the number of tokens in that sentence and $N - dim$ is the number of dimensions in the embedding space. Then, it applies mean pooling so that the resulting vector has dimensions $[1, N - dim]$ to generate fixed-size sentence embeddings. The algorithm is supervised to minimize the cosine distance of similar sentences and maximize the distance of dissimilar ones.

5.2 ClusterSum Extractive Stage

In order to explore the use of semantic representations to select sentences on a multi-document extractive summarization scenario, we developed ClusterSum. The algorithm starts by segmenting the set of input documents into a set of sentences, that are then grouped in a single set, $\mathbf{O}$, as it was for the TF-IDF extractive stage,

$$\mathbf{O} = (Sent_{11}, ..., Sent_{1N(1)}, ..., Sent_{D1}, ..., Sent_{DN(D)}). \tag{5.1}$$

Recall that the extractive stage divides documents Doc_i from $\mathbf{d} = \{Doc_1, ..., Doc_D\}$, D is the number of documents, into sentences: $Doc_i = (Sent_{i1}, ..., Sent_{iN(i)})$. $Sent_{ij} \in Doc_i$ is a sentence with about 100 words, $j \in [1 \dots N(i)]$, and $N(i)$ is the number of sentences in $Doc_i \in \mathbf{d}$, which may vary from document to document.

Then the framework generates sentence embeddings for all input sentences $Sent_{ij} \in \mathbf{O}$. We follow the SBERT methodology to generate sentence embeddings and apply a pre-trained Transformer encoder followed by mean pooling. This step generates the set

$$\mathbf{E} = (Emb_{11}, ..., Emb_{1N(1)}, ..., Emb_{D1}, ..., Emb_{DN(D)}), \tag{5.2}$$

where Emb_{ij} is the sentence embedding associated to sentence $Sent_{ij}$, and $|\mathbf{O}| = |\mathbf{E}|$.

By applying the mean pooling, the resulting vectors Emb_{ij} have a fixed size $[1, N - dim]$, independently of the input sentence size. The encoder architecture will define the $N - dim$ value, which for current implementations of Transformers is commonly 512, 768, or 1024. The framework is model agnostic: one might choose any encoder as the sentence embedding generator. Nevertheless, applying a pre-trained encoder is preferable since it should output embeddings where distances are more semantically meaningful, as explained before.

In order to detect equivalent sentences, we cluster similar embeddings from the set $\mathbf{E}$ by applying the algorithm K-medoids. K-medoids is a well-known clustering algorithm that iteratively finds ultra-spherical clusters given a desired number of clusters. Since it is a text-book clustering method, we will not extend the explanation about how the algorithm works, but one can refer to Park and Jun (2009) for further information.

Here we take advantage of the fact that the ultra-spherical centroids are always actual points of the set, a.k.a embeddings that are central to a cluster. So, we define the number of clusters as L (the number of sentences to extract) and choose the sentences associated with the centroid embeddings as the extractive summary sentences. Finally, the we order medoid sentences by the size of their respective clusters in number of sentences (clusters with more sentences first). The reasoning behind this cluster ordering is that subjects mentioned more often on the input sentences should be more central to the topic.

Figure 9 illustrates the ClusterSum extractive summarization process.

Since the output sentence embeddings are high-dimensional, we also tested applying dimension reduction techniques to avoid the "curse of dimensionality" when clustering. The curse of dimensionality is the phenomenon where data becomes increasingly sparse when the number of dimensions increases. So, as the number of dimensions grows, distances become less meaningful, and more data points are needed to define a cluster.

We experimented with UMAP (MCINNES et al., 2018) as the dimension reduction technique. UMAP assumes that there are lower-dimensional manifolds in high-

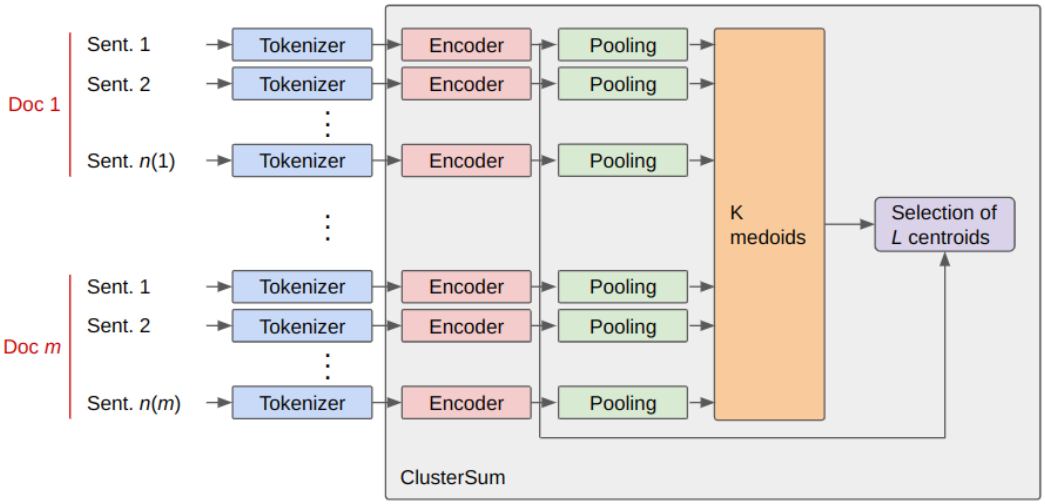
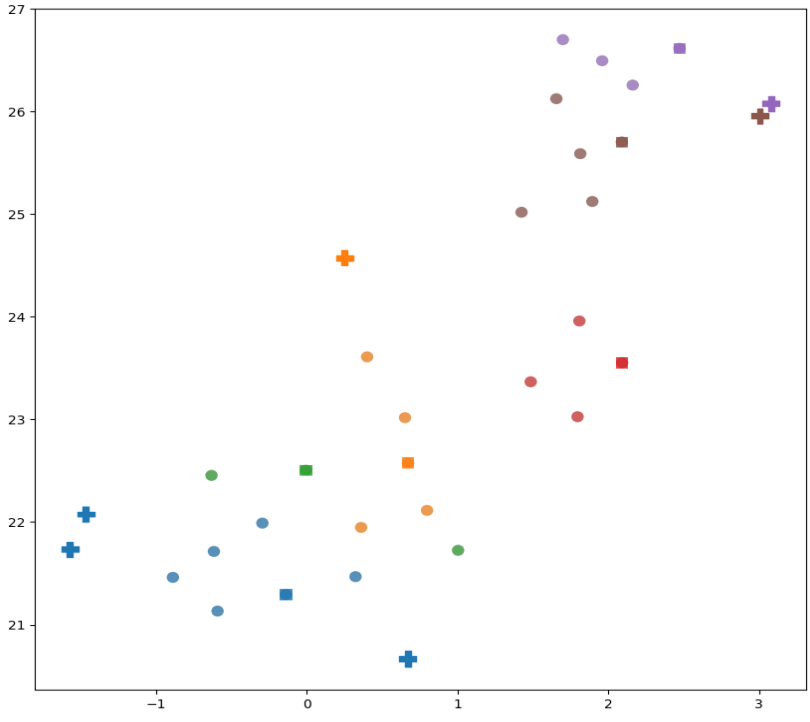


Figure 9 – ClusterSum extractive summarization process. Firstly, tokenized sentences are passed through an encoder followed by a mean pooling operation. The resulting sentence embeddings are clustered with the K-medoids algorithm, and the sentence corresponding to the centroid of each cluster is chosen to compose the set of L sentences.

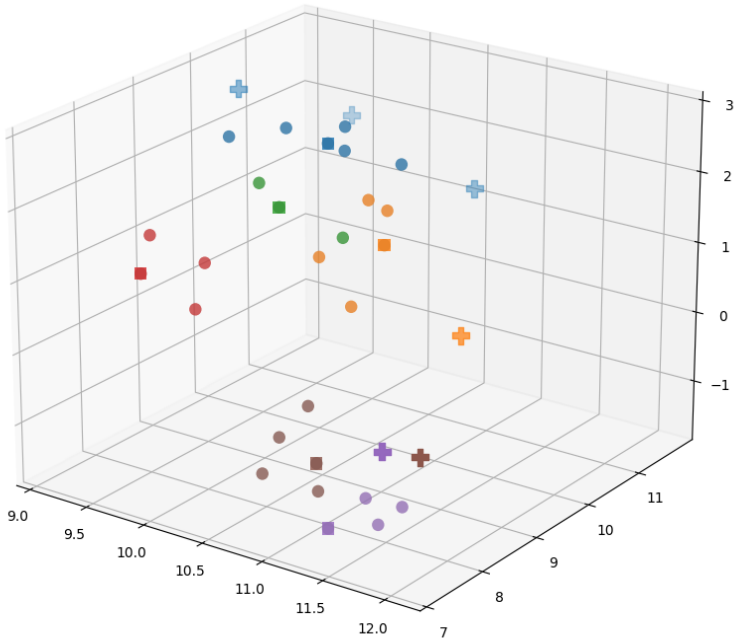
dimensional data and tries to project the original data points to preserve local structures in the transformed space. The algorithm is reported to be effective for clustering because it preserves distances of near data points. Before the clustering stage, we tested to reduce $N - dim$ from 768 to 30 with UMAP.

This way, we achieved an unsupervised extractive summarization algorithm that is model-independent and can easily be applied to any language with available pre-trained encoders. To illustrate the rationale behind the summarization process, Figure 10 shows projections (a) in 2 and (b) 3 dimensions of sentence embeddings extracted from the Multi-News dataset. Circles are input sentences, crosses are reference summary sentences, and squares are the centroids after running the K-medoids algorithm. Colors represent the clusters to which each point was assigned. We can see from the examples that the summary sentence embeddings (crosses) are spread in the space and often associated with different clusters.

Hence, the algorithm assumes that each summary sentence is associated with a different aspect of the described topic. For the multi-document input scenario, several input sentences would inform the same semantic content and thus be near the summary sentences in the embedding space, forming a cluster. We use the centrality of centroid embeddings as a heuristic of relevance to infer which input sentence to extract without prior knowledge of the target summary sentences. On the other hand, picking one sentence from each cluster ensures maximum coverage on the algorithm.



(a) A 2-d projection of sentence embeddings.



(b) A 3-d projection of sentence embeddings.

Figure 10 – Multi-news sentence embeddings projections after dimension reduction to 2 (a) and 3 (b) dimensions. Each color represents a cluster as labeled by the K-medoids algorithm. The crosses represent summary sentences, circles are input sentence embeddings, and squares are input sentence embeddings labeled as cluster centroids by the K-medoids algorithm. One can notice that summary embeddings (crosses) are spread in the representations.

5.3 ClusterSum Experiments and Results

In order to validate ClusterSum capability to generate comprehensive extractive summaries, we add two more experiments to the previously described for PLSum.

Experiment 4: Since ClusterSum is model-independent and can be applied to languages other than Portuguese, we evaluate it in the benchmark dataset for English MDAS, Multi-News (FABBRI et al., 2019). We chose Multi-News as it was extensively applied in the community and have similar characteristics to BrWac2Wiki, as is shown in Table 8. For instance, Multi-News has a similar number of input documents and target size in tokens. By doing so, we can compare ClusterSum results with state-of-the-art summarization algorithms in the literature.

In this experiment, we apply SBERT (REIMERS; GUREVYCH, 2019) as the sentence embedding generator and extract a fixed number of sentences $L = 7$.

Table 8 – Characteristics of MultiNews dataset in percentiles. Input and output sizes are in the number of words.

Percentile (%)	20	40	60	80	100
Nº of input documents	2.0	2.0	3.0	3.0	10.0
Nº of sentences per input	21.0	33.0	45.0	67.0	10758.0
Nº of input tokens	827.0	1293.0	1843.0	2780.0	522945.0
Nº of target tokens	93.0	93.0	93.0	180.0	283.0

Experiment 5: In Experiment 5, we test ClusterSum as the extractive stage of PLSum and apply it to the BrWac2Wiki dataset. Here, we define the pooled output of BERTimbau (SOUZA; NOGUEIRA; LOTUFO, 2020) as the sentence embedding and use $L = 5$ as the fixed number of sentences to extract. Similarly to the previous experiments with BrWac2Wiki, we apply the abstractive framework of PLSum with ClusterSum and PTT5 ($J = 512$). Finally, we report the results of TF-IDF alone and TF-IDF + PTT5 for comparison.

For both experiments, we report the minimum, mean, and maximum f-measure (F) of ROUGE 1, 2, and L on unseen samples from the datasets. We apply a bootstrap resampling on the test set of the dataset with 1000 samples, following the same methodology as Experiments 1, 2, and 3.

The results of **Experiment 4** are reported in Table 9, where TF-IDF, TextRank and Cheating are the same algorithms described in Section 4.1, "ClusterSum" is the version of

Table 9 – ROUGE scores for state of the art summarization algorithms and ClusterSum in the benchmark summarization dataset Multi-News. In this test, we set the number of sentences to extract $L = 7$ (each sentence with a maximum of 100 tokens). SBERT (REIMERS; GUREVYCH, 2019) was utilized as the sentence embedding generator for ClusterSum. The first group (above the line) are extractive-only models, while the second are abstractive models. We highlight the best results in bold.

Model	**R1 F (%)**	**R2 F (%)**	**RL F (%)**
Random (lower bounds)	37.9 [37.6, 38.1]	11.4 [11.2, 11.6]	17.3 [17.2, 17.5]
TF-IDF (our run)	37.8 [37.6, 38.1]	13.2 [12.9, 13.4]	17.7 [17.5, 17.8]
TextRank (our run)	36.4 [36.2, 36.6]	10.3 [10.1, 10.5]	17.1 [17.0, 17.3]
ClusterSum (ours)	39.4 [39.2, 39.6]	12.5 [12.2, 12.7]	18.3 [18.1, 18.4]
ClusterSum + UMAP (ours)	40.9 [40.7, 41.1]	12.5 [12.3, 12.7]	18.8 [18.6 , 19.0]
MatchSum	**46.2**	**16.5**	-
Cheating (upper bounds)	**46.2 [45.9, 46.6]**	**23.6 [23.3, 23.9]**	**21.4 [21.2, 21.6]**
Graph-ED	49.0	19.0	24.0
PRIMERA	**49.9**	**21.1**	**25.9**

the extractive summarization algorithm without dimension reduction before the clustering and "ClusterSum + UMAP" is the version with UMAP. MatchSum (ZHONG et al., 2020) is a state of the art supervised extractive summarization algorithm on the literature, while Graph-ED (PASUNURU et al., 2021), and PRIMERA (XIAO et al., 2022) are state of the art abstractive summarization algorithms[2].

The results show that the extractive-only test with ClusterSum had a slightly better result than TF-IDF on both versions (with and without UMAP). The best result on ROUGE 1 F was with ClusterSum + UMAP (3.1 % and 1.1 %, above TF-IDF for ROUGE 1 and 2 respectively), while on ROUGE 2 F both versions of ClusterSum had simmilar results, 0.7 % bellow TF-IDF.

While ClusterSum had worse results than the algorithms MatchSum, Graph-ED, and PRIMERA, we highlight that all three state of the art algorithms from the literature are supervised, and ClusterSum depends only on self-supervised encoders.

In addition, both Graph-ED and PRIMERA have limitations regarding the input size, given that the computational complexity scales with the number of concatenated tokens. ClusterSum, on the other hand, can process any number of sentences without significantly increasing computational complexity.

The results of **Experiment 5** are reported in Table 10. The extractive-only and abstractive summarization algorithms are separated by the line in the middle (extractive above and abstractive bellow).

[2]Results for state of the art algorithms do not have a range of values because they were extracted as reported in the respective articles for Multi-News dataset

Table 10 – ROUGE scores for state of the art summarization algorithms and ClusterSum in the BrWac2Wiki dataset. In this test, we set the number of sentences to extract $L = 5$ (each sentence with a maximum of 100 tokens). The pooled output of PLSum encoder was utilized as the sentence embedding generator for ClusterSum. The first group (above the line) are extractive-only models, while the second are abstractive ones. The best results for each rouge score are highlighted in bold.

Model	**R1 F (%)**	**R2 F (%)**	**RL F (%)**
TF-IDF	17.8 [17.7, 18.0]	**3.6** [3.5, 3.6]	10.0 [9.9, 10.1]
ClusterSum	**19.6** [19.5, 19.7]	1.7 [1.7, 1.7]	**11.0** [11.0, 11.1]
ClusterSum + UMAP	18.4 [18.25, 18.6]	1.8 [1.7, 1.9]	10.0 [9.9, 10.1]
TF-IDF + PTT5	**32.0** [31.6, 32.3]	**15.0** [14.7, 15.3]	**25.6** [25.2, 26.0]
ClusterSum + PTT5	29.3 [29.0, 29.6]	12.7 [12.4, 13.0]	23.5 [23.2, 23.9]
ClusterSum + UMAP + PTT5	27.9 [27.6, 28.2]	11.3 [11.0, 11.6]	22.2 [21.9, 22.5]

Comparing the extractive-only algorithms, we observe the same trend of Experiment 4 for most ROUGE scores, where ClusterSum had slightly better results than TF-IDF on ROUGE 1 and ROUGE L (1.8 and 1.0 points above TF-IDF). For ROUGE 2, on the other hand, both versions of ClusterSum had underwhelming results, 1.8 % bellow TF-IDF. Also, the version of ClusterSum without UMAP showed better ROUGE 1 and L scores than the version with UMAP (1.2 % and 1.0 %).

We observed a different pattern on the abstractive summarization scenario: The framework version with ClusterSum + PTT5 had lower ROUGE scores than TF-IDF + PTT5 (2.7%, 2.3%, and 2.1% for ROUGE 1, 2, and F, respectively). To better understand these instigating results, we refer to the examples of extractive summaries generated by TF-IDF and ClusterSum in Table 11.

One can notice that the TF-IDF method often extracted small sentences citing the title (see the summaries for “Usina Hidrelétrica de Itaipu” and “Membrana”). Also, upwards sentences tends to be more crucial to the topic. The drawback is that the small sentences are sometimes uninformative, as is shown on the first two sentences for “Membrana” topic. ClusterSum, on the other hand, managed to select diverse sentences for the three samples, and their sizes are overall bigger. Still, ClusterSum do not seem to prioritize the order of sentences by relevance properly, and some times sentences are off-topic.

We tried to tackle two problems observed in TF-IDF: the repetition of similar phrases in the summary and selection of true homonyms or unrelated sentences. From the observation of the summaries, we argue that ClusterSum showed promising results in solving the first problem, that is, the repetition of sentences. That diversity of subjects might also explain why the dense algorithm worked better as an extractive-only summarizer while been worse as an extractive step for the complete framework: The different sentences, out

Table 11 – Examples of extractive summaries generated by TF-IDF and ClusterSum.

TF-IDF	ClusterSum
Usina Hidrelétrica de Itaipu: **(SENT1)** o que é uma usina hidrelétrica ? **(SENT2)** fotografia da usina de itaipu , a maior hidrelétrica das américas projeto da usina hidrelétrica de belo monte. **(SENT3)** 2º usina de itaipu – brasil (14.000mw). **(SENT4)** usina hidreletrica de itaipu-foz do iguacu **(SENT5)** a barragem da hidreletrica de itaipu ja com o reservatorio formado abriu pela 1. vez as comportas de seu vertedouro proporcionando as cerca de 1.000 pessoas que se espalharam nas imediacoes da usina um belo espetaculo visual.	**Usina Hidrelétrica de Itaipu:** **(SENT1)** nos tempos modernos, com a mobilização de organizações não governamentais contra o desmatamento e agressão ao meio ambiente , muitas polêmicas e questionamentos são feitos com relação a construção de novas usinas hidrelétricas ou outras formas de geração de energia. **(SENT2)** outros sim , a depopulação desses diferentes grupos foi dramática **(SENT3)** assim , o primeiro grupo sofreu um decréscimo populacional de 54 no primeiro ano de contato (ocorrido em 1971). **(SENT4)** os contatados posteriormente apresentaram índices de depopulação em torno de 25% (...)
Cálculo: **(SENT1)** de maneira geral poderíamos falar em quatro tipos de cálculo que deveriam ser explorados e exercitados na escola: o cálculo escrito (algoritmos), o cálculo mental exato, o cálculo mental aproximado (estimativas) e o cálculo feito com ferramentas de apoio, das quais a mais comum é a calculadora. **(SENT2)** além disso, o material propicia situações que levam os alunos a usar equilibradamente as várias formas de cálculo: o cálculo escrito; a estimativa; o cálculo mental e o uso de instrumentos como a calculadora. **(SENT3)** 2. cálculo mental e estimativa . **(SENT4)** gostaria que me enviasse, se possível, o cálculo feito para a raiz quadrada do número 20, aquele cálculo sem a calculadora (...)	**Cálculo:** **(SENT1)** para fazer um exercício , é bom se organizar e anotar os dados importantes do exercício . em alguns exercícios de soma dos termos da pg será necessário calcular algum dado que não foi informado diretamente no exercício , como por exemplo a razão q . **(SENT2)** se imaginarmos agora que o ponto se aproxima do ponto , xfx , podemos ver que a corda se aproxima da reta tangente à curva no ponto, xfx. **(SENT3)** quer achar uma porcentagem com a calculadora e não sabe como fazê-lo ? é muito simples , todas as calculadoras incluem um tecla específica para calcular as porcentagens e assim simplificar este cálculo (...)
Membrana: **(SENT1)** composição e propriedades da membrana . **(SENT2)** 2 - transporte através da membrana plasmática : **(SENT3)** a membrana plasmática é : muito fina ; **(SENT4)** as proteínas inseridas na membrana não são fixas : podem deslizar ao longo do plano da membrana . isso confere à membrana plasmática , outra característica importante , a de que a porção lipídica é fluida . **(SENT5)** membrana plasmática cola da web todas as células procariotas e eucariotas apresentam na superfície um envoltório a membrana plasmática também chamada de membrana citoplasmática ounbsp . . . ver o linksemelhante	**Membrana:** **(SENT1)** estabelecem junções comunicantes , espécie de canais de comunicação entre células muito próximas , que possibilitam trocas de pequenas moléculas informacionais . **(SENT2)** as membranas filtrantes advantec mfs são películas plásticas microporosas com tamanho de poros específicos para cada tipo de partícula . são membranas que retêm partículas ou microorganismos maiores que seu tamanho de poro através da interação em superfície . **(SENT3)** é aquele que ocorre contra um gradiente de concentração , ou seja , de um meio menos concentrado para um meio de maior concentração . para tanto , a célula consome energia metabólica . ex : bomba de na + e k + (...)

of context, could confuse the abstractive stage.

Finally, Table 12 shows the summarization time for the candidate extractive summarizers on the BrWac2Wiki test set, with $L = 5$.

TF-IDF is the fastest, 11 and 6 times faster than ClusterSum and ClusterSum + UMAP, respectively, and TextRank was the slower, 26 times slower than TF-IDF. ClusterSum + UMAP is faster than the version without UMAP, even with one more step in its procedure. It happens because fitting the K-medoids algorithm with fewer dimensions is considerably faster.

So, considering that: (1) both versions of ClusterSum slightly failed to surpass the baseline ROUGE scores as a preliminary stage of the abstractive summarization pipeline; and (2) ClusterSum is slower and more complex than the baseline; We conclude that

Table 12 – Average summarization time for each extractive model on the BrWac2Wiki dataset. The summarization time was calculated as the time of applying the models with $L = 5$ on a full run on the test set divided by the number of samples.

Model	**Summ. time (s)**	**Δ**
TF-IDF	0.07 s	–
TextRank	1.87 s	26 x TF-IDF
ClusterSum	0.85 s	11 x TF-IDF
ClusterSum + UMAP	0.46 s	6 x TF-IDF

applying such a dense extractive stage might not be justified for specific abstractive summarization tasks. On the other hand, the model displayed better overall results as an extractive-only algorithm. Also, we observed on the qualitative study that ClusterSum can generate more diverse summaries, what is generally desirable for applications.

6 CONCLUSION

In this work, we discussed several strategies to generate abstractive summaries from multiple documents. We focused our analysis on Brazilian Portuguese, but readers could extrapolate the solutions to other languages, especially other "peripheral" or misrepresented languages in the AI community.

Despite displaying some unreliable information, the final version of our model showed great synthesis capacity and could create lengthy, comprehensive summaries, clearly in the style of Wikipedia. In this way, we contribute a step towards the automatic generation of articles for uncovered wiki topics and the means for future research to tackle the task of MDAS in Portuguese.

The results support the conclusions of previous work that abstractive models can generate better-written summaries than extractive ones when fine-tuned on similar data. The experiments in Section 4.4 corroborate this conclusion. In addition, the algorithm displayed underwhelming results on the CSTNews dataset when applied without fine-tuning. That result shows the model needs fine-tuning on similar data for better performance. Furthermore, we observed that abstractive methods adapted to different styles and performed a "transparent" surface realization on the summary. Another major conclusion of this work is that pre-training is very effective and is more determinant than applying an architecture that handles more text in the input.

For the extractive stage, we observed that TF-IDF and TextRank could not guarantee the coverage and non-redundancy of the retrieved texts. Notably, TF-IDF, the sparse extractive technique that showed better results on ROUGE scores, is biased towards small sentences citing the query.

Our algorithm applying dense semantic representations, ClusterSum, showed slightly better ROUGE scores as an extractive-only summarizer but worse contribution as a step on the complete framework (extractive + abstractive). One hypothesis to explain the contradictory results is that the bias of TF-IDF to smaller sentences directly citing the

title might help it to retrieve more straightforward sentences. Thus, it might be easier for the abstractive stage to interpret its input with TF-IDF.

It is also worth mentioning that automatic metrics such as ROUGE have granularity limitations when evaluating summaries, as is shown in Schluter (2017). Thus, it is inconclusive to compare minor differences in scores, like the ones obtained for TF-IDF + PTT5 and ClusterSum + PTT5 versions. Since automatically generated summaries have fewer grammar mistakes after the latest improvements on language models, their evaluation gets more subjective. So defining better metrics and summary evaluation procedures is crucial to further advance the field.

Finally, the qualitative analysis indicates that the dense algorithm selects more diverse sentences than the sparse ones. In this sense, our experiments showed that exploring dense semantic representations of sentences is viable to improve the coverage of extractive summaries. We argue that an advantage of using dense strategies for the extractive stage is that they share similar representations to the state-of-the-art Transformer abstractive stage. So, algorithms could benefit from generating sentence embeddings once on the extractive stage, and then applying a simpler decoder-only generative model for the second part. Also, developing end2end training strategies for MDAS might improve the results of the dense extractive stage.

In addition, we couldn't fail to comment on the big star of the moment in the NLP area launched in November 2022 by OpenAI: ChatGPT. This freely available model stands out for its detailed responses and articulate answers across many domains of knowledge. Thus, we made a small comparative analysis of ChatGPT with our proposal, that can be found in Appendix A of this document. While we are entering a new phase of Artificial Intelligence, some technical details regarding ChatGPT still need to be clarified before it is widely used to avoid negative results such as spreading misinformation. We have to recognize that ChatGPT is an advanced chat-bot that has the potential to make people's lives a lot easier, when used well.

To conclude, while the state-of-the-art for MDAS has significantly improved with new techniques, there is much to be done to have reliable and accurate summarization algorithms. In this sense, there are several exciting themes for future research. One of them is addressing the factual inaccuracy of such abstractive models. A straightforward solution is using mixed extractive and abstractive techniques, such as Pointer-generator (SEE; LIU; MANNING, 2017) so that the model can copy crucial information from the input. Also, increasing the training dataset can lessen the effect of biases toward com-

monplaces. Neuro-symbolic strategies are also promissing, because models could benefit from logical formulations, as is often the case for humans. For the Brazilian Portuguese MDAS scenario, another future possibility is the application of pre-training in seq2seq models with a receptive field superior to T5, such as Longformer, which can lead to more complete summaries. For the extractive stage, we believe that developing supervised methods based on sentence embeddings might significantly improve the current results in Portuguese datasets, as it was already shown for English on MatchSum (ZHONG et al., 2020).

REFERENCES

ALCOFORADO, A.; FERRAZ, T. P.; GERBER, R.; BUSTOS, E.; OLIVEIRA, A. S.; VELOSO, B. M.; SIQUEIRA, F. L.; COSTA, A. H. R. Zeroberto: Leveraging zero-shot text classification by topic modeling. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2022. p. 125–136.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. In: BENGIO, Y.; LECUN, Y. (Ed.). *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. [s.n.], 2015. Available at: ⟨http://arxiv.org/abs/1409.0473⟩.

BANERJEE, S.; MITRA, P.; SUGIYAMA, K. Multi-document abstractive summarization using ILP based multi-sentence compression. *IJCAI International Joint Conference on Artificial Intelligence*, v. 2015-Janua, n. Ijcai, p. 1208–1214, 2015. ISSN 10450823.

BELTAGY, I.; PETERS, M. E.; COHAN, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

BING, L.; LI, P.; LIAO, Y.; LAM, W.; GUO, W.; PASSONNEAU, R. J. Abstractive multi-document summarization via phrase selection and merging. *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, v. 1, p. 1587–1597, 2015.

CAÇÃO, F. N.; JOSÉ, M. M.; OLIVEIRA, A. S.; SPINDOLA, S.; COSTA, A. H. R.; COZMAN, F. G. Deepagé: answering questions in portuguese about the brazilian environment. In: SPRINGER. *Brazilian Conference on Intelligent Systems*. [S.l.], 2021. p. 419–433.

CARBONELL, J.; GOLDSTEIN, J. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 1998. p. 335–336.

CARMO, D.; PIAU, M.; CAMPIOTTI, I.; NOGUEIRA, R.; LOTUFO, R. Ptt5: Pretraining and validating the t5 model on brazilian portuguese data. *arXiv preprint arXiv:2008.09144*, 2020.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*. [S.l.: s.n.], 2014.

CHOPRA, S.; AULI, M.; RUSH, A. M. Abstractive sentence summarization with attentive recurrent neural networks. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [S.l.: s.n.], 2016. p. 93–98.

CHU, E.; LIU, P. Meansum: a neural model for unsupervised multi-document abstractive summarization. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2019. p. 1223–1232.

COAVOUX, M.; ELSAHAR, H.; GALLÉ, M. Unsupervised aspect-based multi-document abstractive summarization. In: *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. [S.l.: s.n.], 2019. p. 42–47.

DANG, H. T.; OWCZARZAK, K. et al. Overview of the tac 2008 update summarization task. In: *TAC*. [S.l.: s.n.], 2008.

DENKOWSKI, M.; LAVIE, A. Meteor universal: Language specific translation evaluation for any target language. In: *Proceedings of the ninth workshop on statistical machine translation*. [S.l.: s.n.], 2014. p. 376–380.

DEVLIN, J.; CHANG, M.; LEE, K.; TOUTANOVA, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. Available at: ⟨http://arxiv.org/abs/1810.04805⟩.

FABBRI, A. R.; LI, I.; SHE, T.; LI, S.; RADEV, D. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2019. p. 1074–1084.

GENEST, P.-E.; LAPALME, G. Framework for abstractive summarization using text-to-text generation. In: *Proceedings of the workshop on monolingual text-to-text generation*. [S.l.: s.n.], 2011. p. 64–73.

GERANI, S.; MEHDAD, Y.; CARENINI, G.; NG, R.; NEJAT, B. Abstractive summarization of product reviews using discourse structure. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1602–1613.

GRAVES, A. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

GUPTA, S.; GUPTA, S. Abstractive summarization: An overview of the state of the art. *Expert Systems with Applications*, Elsevier, v. 121, p. 49–65, 2019.

HERMANN, K. M.; KOČISKÝ, T.; GREFENSTETTE, E.; ESPEHOLT, L.; ESPEHOLT, L.; KAY, W.; SULEYMAN, M.; BLUNSOM, P. Teaching Machines to Read and Comprehend NIPS 2015. *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS'15)*, p. 1693–1701, 2015.

HOCHREITER, S.; SCHMIDHUBER, J. Long Short-Term Memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, nov 1997. ISSN 0899-7667.

JONES, K. S. et al. Automatic summarizing: factors and directions. *Advances in automatic text summarization*, Cambridge, MA: MIT Press, p. 1–12, 1999.

JORGE, M. L. d. R. C. *Sumarização automática multidocumento: seleção de conteúdo com base no Modelo CST (Cross-document Structure Theory)*. Tese (Doutorado) — Universidade de São Paulo, 2010.

KARPUKHIN, V.; OGUZ, B.; MIN, S.; LEWIS, P.; WU, L.; EDUNOV, S.; CHEN, D.; YIH, W.-t. Dense passage retrieval for open-domain question answering. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. p. 6769–6781. Available at: ⟨https://www.aclweb.org/anthology/2020.emnlp-main.550⟩.

KINGSBURY, P. R.; PALMER, M. From treebank to propbank. In: CITESEER. *LREC*. [S.l.], 2002. p. 1989–1993.

LEBANOFF, L.; SONG, K.; LIU, F. Adapting the neural encoder-decoder framework from single to multi-document summarization. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, p. 4131–4141, 2018.

LEIXO, P. et al. *CSTNews: um corpus de textos jornalísticos anotados segundo a Teoria Discursiva Multidocumento CST (Cross-document Structure Theory)*. São Carlos, SP, Brazil, 2008.

LIN, C.-Y.; OCH, F. Looking for a few good metrics: Rouge and its evaluation. In: *Ntcir Workshop*. [S.l.: s.n.], 2004.

LIN, H.; NG, V. Abstractive Summarization: A Survey of the State of the Art. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 33, n. 4, p. 9815–9822, 2019. ISSN 2159-5399.

LIU, P. J.; SALEH, M.; POT, E.; GOODRICH, B.; SEPASSI, R.; KAISER, L.; SHAZEER, N. Generating wikipedia by summarizing long sequences. In: *International Conference on Learning Representations*. [s.n.], 2018. Available at: ⟨https://openreview.net/forum?id=Hyg0vbWC-⟩.

MARCUS, M.; KIM, G.; MARCINKIEWICZ, M. A.; MACINTYRE, R.; BIES, A.; FERGUSON, M.; KATZ, K.; SCHASBERGER, B. The penn treebank: annotating predicate argument structure. In: *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*. [S.l.: s.n.], 1994.

MCINNES, L.; HEALY, J.; SAUL, N.; GROSSBERGER, L. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, v. 3, n. 29, p. 861, 2018.

MIHALCEA, R.; TARAU, P. Textrank: Bringing order into text. In: *Proceedings of the 2004 conference on empirical methods in natural language processing*. [S.l.: s.n.], 2004. p. 404–411.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2013. p. 3111–3119.

NENKOVA, A.; PASSONNEAU, R. J. Evaluating content selection in summarization: The pyramid method. In: *Proceedings of the human language technology conference of the north american chapter of the association for computational linguistics: Hlt-naacl 2004*. [S.l.: s.n.], 2004. p. 145–152.

OLIVEIRA, A.; COSTA, A. Plsum: Generating pt-br wikipedia by summarizing multiple websites. In: *Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional*. Porto Alegre, RS, Brasil: SBC, 2021. p. 751–762. ISSN 2763-9061. Available at: ⟨https://sol.sbc.org.br/index.php/eniac/article/view/18300⟩.

OLIVEIRA, A.; COSTA, A.; HRUSCHKA, E. A framework for multi-document extractive summarization of reviews with aspect-based sentiment analysis. In: SBC. *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*. [S.l.], 2020. p. 471–482.

OVER, P. An Introduction to DUC 2004 Intrinsic Evaluation of Generic New Text Summarization Systems. *Proceedings of the Document Understanding Conference*, 2004.

OVER, P.; DANG, H.; HARMAN, D. DUC in context. *Information Processing and Management*, v. 43, n. 6, p. 1506–1520, 2007. ISSN 03064573.

PAGE, L.; BRIN, S.; MOTWANI, R.; WINOGRAD, T. *The PageRank citation ranking: Bringing order to the web.* [S.l.], 1999.

PAIOLA, P. H.; ROSA, G. H. de; PAPA, J. P. Deep learning-based abstractive summarization for brazilian portuguese texts. In: SPRINGER. *Intelligent Systems: 11th Brazilian Conference, BRACIS 2022, Campinas, Brazil, November 28–December 1, 2022, Proceedings, Part II*. [S.l.], 2022. p. 479–493.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. Bleu: a method for automatic evaluation of machine translation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 40th annual meeting on association for computational linguistics*. [S.l.], 2002. p. 311–318.

PARK, H.-S.; JUN, C.-H. A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, Elsevier, v. 36, n. 2, p. 3336–3341, 2009.

PASCHOAL, A. F. A.; PIROZELLI, P.; FREIRE, V.; DELGADO, K. V.; PERES, S. M.; JOSé, M. M.; NAKASATO, F.; OLIVEIRA, A. S.; aO, A. A. F. B.; COSTA, A. H. R.; COZMAN, F. G. Pirá: A bilingual portuguese-english dataset for question-answering about the ocean. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2021. (CIKM '21), p. 4544–4553. ISBN 9781450384469. Available at: ⟨https://doi.org/10.1145/3459637.3482012⟩.

PASUNURU, R.; LIU, M.; BANSAL, M.; RAVI, S.; DREYER, M. Efficiently summarizing text and graph encodings of multi-document clusters. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [S.l.: s.n.], 2021. p. 4768–4779.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543.

RAFFEL, C.; SHAZEER, N.; ROBERTS, A.; LEE, K.; NARANG, S.; MATENA, M.; ZHOU, Y.; LI, W.; LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, v. 21, p. 1–67, 2020.

RAMOS, J. et al. Using tf-idf to determine word relevance in document queries. In: CITESEER. *Proceedings of the first instructional conference on machine learning*. [S.l.], 2003. v. 242, n. 1, p. 29–48.

REIMERS, N.; GUREVYCH, I. Sentence-bert: Sentence embeddings using siamese bert-networks. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. [S.l.: s.n.], 2019. p. 3982–3992.

RIBALDO, R.; AKABANE, A. T.; RINO, L. H. M.; PARDO, T. A. S. Graph-based methods for multi-document summarization: exploring relationship maps, complex networks and discourse information. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2012. p. 260–271.

RUSH, A. M.; CHOPRA, S.; WESTON, J. A neural attention model for sentence summarization. *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, p. 379–389, 2015.

SCHLUTER, N. The limits of automatic summarisation according to rouge. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. [S.l.], 2017. p. 41–45.

SEE, A.; LIU, P. J.; MANNING, C. D. Get to the point: Summarization with pointer-generator networks. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. [S.l.: s.n.], 2017. p. 1073–1083.

SILVEIRA, S. B.; BRANCO, A. Enhancing multi-document summaries with sentence simplification. In: CITESEER. *Proceedings on the International Conference on Artificial Intelligence (ICAI)*. [S.l.], 2012. p. 1.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. BERTimbau: pretrained BERT models for Brazilian Portuguese. In: *9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)*. [S.l.: s.n.], 2020.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 3104–3112.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 5998–6008.

WAGNER, J.; WILKENS, R.; IDIART, M.; VILLAVICENCIO, A. The brwac corpus: A new open resource for brazilian portuguese. In: . [S.l.: s.n.], 2018. p. 4339–4344.

XIAO, W.; BELTAGY, I.; CARENINI, G.; COHAN, A. Primera: Pyramid-based masked sentence pre-training for multi-document summarization. In: *Proceedings of the*

60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). [S.l.: s.n.], 2022. p. 5245–5263.

ZHONG, M.; LIU, P.; CHEN, Y.; WANG, D.; QIU, X.; HUANG, X.-J. Extractive summarization as text matching. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2020. p. 6197–6208.

APPENDIX A – COMPARISON TO CHATGPT FOR SUMMARIZATION

Recently, ChatGPT, a multi-purpose chat bot was made available on the web by the company OpenAi. Users can openly trial the algorithm, applying it for several use cases and languages.

With the right prompt, the algorithm can also generate descriptions of topics or summaries for extracts of texts. While the company did not publish an specific article yet, their website describe the model as a fine-tune of GPT3.5, a Transformer-based language model trained to follow instructions in a prompt. The fine-tune features Reinforcement Learning and human-in-the-loop concepts.

Given its success and remarkable capability for several tasks, we compared it to PLSum in generating descriptions for some topics and show the results on Table 13.

While we can't make systematic comparisons on multiple samples, we observed that ChatGPT's summaries are generally better, as they have less grammar mistakes and false information. It is noticeable how ChatGPT can generate long descriptions that rarely display syntactic and semantic errors. Thus, the prompt-answering learning technique they propose is promising for general-use language models. Despite that, it is worth pointing out some drawbacks.

First, the lack of detailed information on the architecture, training and test on open datasets hardens the comparison to other models. The website description indicates the model architecture have billions of parameters. Sources on the internet claim it has 175 billion parameters and was trained on roughly 500 billion tokens. Also, an estimated 10,000 GPUs were used to train the model [1], showing the enormous work of the team (and investments in resources) to make this chat-bot available. Evidently, it is a considerably

[1] https://www.fierceelectronics.com/sensors/chatgpt-runs-10k-nvidia-training-gpus-potential-thousands-more

larger model than the version of PTT5 we tested on, PTT5-base, with around 220M parameters.

Despite that, we see on the examples of Table 13 that ChatGPT still outputs wrong or misleading information, similarly to other language models. For instance, on the first example we see that it mixed descriptions for a generic membrane and cell's membranes, while PLSum chose to focus on the cell's membrane. This problem might be related to the retrieval of synonyms on the models knowledge base, although it is unclear even if the model uses a knowledge base during the generation of sentences.

It is evident that adjustments to the prompt could alleviate this problem, though. For instance, you could specify what sort of membrane you want to generate a summary for. However, this kind of human interaction would also limit its utilization for massive generation of texts for collaborative projects, such as Wikipedia.

We also see another example of factual mistake on the first paragraph of Mario de Andrade biography (a check on the actual wiki page shows he never studied in Europe, while the algorithm says he did).

Finally, another drawback of ChatGPT is that the model does not show it's sources. That could cause copy-write and fact-checking issues, since you cannot know if the model's output have content from several sources or a copy of just one. In fact, you cannot even check if the sources the model is trained on are copy-write protected. Displaying references is a fundamental aspect of Wikipedia, as is described in its policies and guideline page [2]. The extractive stage of PLSum, on the other hand, makes it easy to check sources, and our test webpage shows from what documents the summary is generated.

So, while ChatGPT has groundbreaking results and capabilities, it still suffers from the same difficulties other language models have with factual mistakes, and could be more transparent about it's outputs. A promising advance to alleviate these problems could be the exploration of explainability models focused on natural language generation.

[2] Available at: https://en.wikipedia.org/wiki/Wikipedia:Policies_and_guidelines, visited in 10/03/2023.

Table 13 – Examples of extractive summaries generated by TF-IDF and ClusterSum.

PLSum	**ChatGPT**
Membrana: A membrana é um envoltório de membranas celulares que envolve as células procariotas e eucariotas. A membrana possui uma camada de lipídios esteróides que se encontram em uma solução de solução de membrana. A membrana pode ser formada por uma camada lipídica que se encontra em um plano de membrana ou em uma camada externa de membrana que é a camada externa da membrana. A camada interna da membrana é composta por lipídeos esteroides que são transportados através da membrana para o interior da célula.	**explique o conceito de membrana:** A membrana é uma estrutura fina e flexível que separa dois ambientes diferentes, geralmente com diferentes concentrações de substâncias. A membrana é composta principalmente de lipídios (gorduras) e proteínas que se organizam em uma bicamada lipídica, com as cabeças hidrofílicas (atraídas pela água) voltadas para fora e as caudas hidrofóbicas (repelidas pela água) voltadas para dentro. Essa bicamada lipídica serve como uma barreira para a maioria das moléculas, controlando seletivamente o que entra e sai da célula ou organela. As proteínas associadas à membrana desempenham uma variedade de funções, incluindo transporte de substâncias através da membrana, ancoragem da membrana a outras estruturas, reconhecimento de outras células e sinais químicos, e catalisar reações bioquímicas.
Feudalismo: O feudalismo foi um modo de organização social e política baseado nas relações servo contratos servis. Tem suas origens na decadência do império romano. O sistema feudal se caracterizou pela concessão de feudos, quase sempre em forma de terras e trabalho, em troca de proteção política e militar. esse contrato feito era selado por um juramento de fidelidade.	**O que foi feudalismo:** O feudalismo foi um sistema político, social e econômico que se desenvolveu na Europa Ocidental durante a Idade Média, entre os séculos V e XV. Esse sistema foi caracterizado pela descentralização do poder, onde a terra era a base da riqueza e do poder político, e pela relação de suserania e vassalagem entre senhores feudais e seus vassalos.
Mario de Andrade: Mário Raul de Moraes Andrade (são paulo, 9 de outubro de 1893 rio de janeiro, 10 de agosto de 1945) foi um poeta, ensaísta, crítico literário, poeta e ensaísta brasileiro.	**Quem foi Mario de Andrade:** Mário de Andrade (1893-1945) foi um importante escritor, poeta, musicólogo e pesquisador brasileiro, considerado um dos grandes expoentes do modernismo brasileiro. Ele nasceu em São Paulo e estudou filosofia, história e música em universidades brasileiras e europeias. Mário de Andrade foi um dos fundadores do modernismo no Brasil, movimento que buscava renovar a literatura e a cultura brasileira, valorizando as raízes e a diversidade cultural do país. Entre suas obras mais famosas estão "Macunaíma", romance que combina elementos do folclore brasileiro com a modernidade urbana, e "Pauliceia Desvairada", livro de poemas que retrata a cidade de São Paulo em transformação. Além de sua produção literária, Mário de Andrade também se destacou como pesquisador da cultura popular brasileira e da música, tendo compilado diversas canções populares e tradicionais do Brasil. Ele foi um defensor do patrimônio cultural do país e um crítico da elite cultural brasileira que ignorava a diversidade e riqueza da cultura popular. Mário de Andrade faleceu em 1945, em São Paulo, deixando uma obra literária e cultural marcante na história do Brasil. Ele é considerado um dos maiores nomes da literatura brasileira e um dos principais representantes do modernismo no país.