Maria João Simões Costa

Automatic Summarization for the Generation of Slides

1 2 9 0 UNIVERSIDADE D COIMBRA

1 2 9 0

UNIVERSIDADE D
COIMBRA

Maria João Simões Costa

Automatic Summarization for the Generation of Slides

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Hugo Gonçalo Oliveira and Hugo Amaro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2022

Faculty of Sciences and Technology

Department of Informatics Engineering

Automatic Summarization for the Generation of Slides

Maria João Simões Costa

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Hugo Gonçalo Oliveira and Hugo Amaro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2022

UNIVERSIDADE D
COIMBRA

Abstract

Technology is becoming increasingly important in today's world, with applications in practically every aspect of people's lives. This is the case in education, where slide shows are one of the most widely used tools during the presentation of specific topics. Creating them, on the other hand, can be a complex and time consuming task, since before presenting the results in slides, it is necessary to read and summarize several documents related to a given subject. Artificial Intelligence methods such as machine learning and natural language processing can be used to automatically create slide decks, allowing teachers and trainers in general to make better use of their time by only having to delete or add certain elements rather than having to start from scratch.

This thesis provides an overview of several different methods used in studies for the automatic generation of presentation slides, and it also reports on a study and comparison of several summarization methods of two types: abstractive and extractive. Some extractive methods are mentioned in the state of the art, while others were only previously used for summarization and are tested in this work in a slide generation context. The abstractive methods, which present two approaches to document summarization—one that summarises the entire text and the other that summarises individual sections—have never before been used for slide generation. Both supervised and unsupervised extractive methods are used. The unsupervised extractive methods and one of the abstractive methods are evaluated in both English and Portuguese. Furthermore, three datasets are used for the experiments: two are composed of pairs of documents and slides, while the other was created specifically for this study and it is composed of Wikipedia articles. These datasets were used to evaluate all the investigated methods automatically using three different metrics. After that, slide decks of Wikipedia articles were created and evaluated by humans.

The results tell us that there is not a single best method. The chosen method will vary depending on the context in which it is used. However, the people that evaluated the slides considered them, independently of the given method, a good starting point to create the final slide presentation, which is the main goal of this project. So, even though there is not a method that can be considered the best for every text summarization, this thesis presents the advantages and limitations of several methods, which will help in the creation of future summaries and, consequently, in the automation of the creation of slide decks, which is currently completely manual.

Keywords

Summarization; Automatic Generation of Slides; Extractive Methods; Abstractive Methods; Natural Language Processing; Transformers

Resumo

A tecnologia está a tornar-se cada vez mais importante no mundo de hoje, com aplicações em praticamente todos os aspectos da vida das pessoas. Isto é o caso da educação, onde slides de apresentação são uma das ferramentas mais utilizadas para demonstrar facilmente certos tópicos. Por outro lado, criá-los pode ser uma tarefa complexa e demorada; é necessário ler e resumir vários documentos relacionados a um determinado assunto antes de apresentar os resultados em slides. Métodos de inteligência artificial, como aprendizagem automática e processamento de linguagem natural, podem ser usados para criar conjuntos de slides automaticamente, permitindo que os professores usem melhor seu tempo, bastando excluir ou adicionar determinados elementos nos slides, em vez de começar do zero.

Esta tese fornece uma visão geral de vários métodos diferentes usados em estudos para a geração automática de slides de apresentação, e também relata um estudo e comparação de vários métodos de sumarização de dois tipos: abstrativos e extrativos. Alguns métodos extrativos são mencionados no estado da arte, enquanto outros foram usados anteriormente apenas para sumarização e são testados neste trabalho em um contexto de geração de slides. Os métodos abstrativos, que apresentam duas abordagens para a sumarização de documentos – uma que resume todo o texto e outra que resume seções individuais – nunca foram usados para geração de slides. Métodos extrativos supervisionados e não supervisionados são usados. Os métodos extrativos não supervisionados e um dos métodos abstrativos são avaliados em inglês e português. Além disso, três datasets são utilizados para as experiências: dois são compostos por pares de documentos e slides, enquanto o outro foi criado especificamente para este estudo e é composto por artigos da Wikipédia. Esses datasets foram usados para avaliar todos os métodos investigados automaticamente usando três métricas diferentes. Depois disso, os slides dos artigos da Wikipedia foram criados e avaliados por humanos.

Os resultados dizem-nos que não existe um método melhor que os outros. O método escolhido depende do contexto em que é usado. No entanto, as pessoas que avaliaram os slides consideraram-nos, independentemente do método fornecido, um bom ponto de partida para criar a apresentação de slides final, sendo que isso é o principal objetivo deste projeto. Assim, embora não exista um método que possa ser considerado o melhor para cada sumário, esta tese apresenta as vantagens e limitações de diversos métodos, que ajudarão na criação de sumários futuros e, consequentemente, na automatização da criação de decks de slides, que atualmente é totalmente manual.

Palavras-Chave

Sumarização; Geração Automática de Slides; Métodos Extrativos; Métodos Abstrativos; Processamento de linguagem natural; Transformadores

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, technology plays a very important part in everyone's lives. More and more technology is used as a resource in several areas. This happens, for example, in education, that resorts to technology to present more engaging lessons, through tools like slide presentations. Teachers can use this tool to present a subject in a more visual way, allowing students to more easily follow what is being presented. However, constructing a slide presentation can be a very time and effort consuming task. Therefore, this work explores the automatic creation of slide decks based on documents, for at least giving teachers or others a good place to start when creating the final presentation.

The section below gives a more detailed description of slideshows, why they are widely used (section 1.1) and why developing a tool that creates them automatically may be valuable. The project's goals are more discussed in section 1.2, and the proposed approach is outlined in section 1.3. The project's main contributions are in section 1.4, while the context is presented in section 1.5, and the introduction to all subsequent chapters of this thesis is given in section 1.6.

1.1 Motivation

Slideshows are a support tool used for presenting a certain theme or subject, serving as a guide that helps the speaker and their audience to follow what is presented. While there is a good, widely used selection of tools to create slideshows, like Microsoft PowerPoint[1], Prezi[2], etc, they only help in making aesthetically appealing slides with the use of themes or templates, whereas adding contents is up to the author of the presentation. On the other hand, tools for the automatic generation of the slides' textual content are still in their infancy, also due to the task's complexity. In fact, selecting the information that best covers all the key topics of a certain document or theme can be a challenging and time-consuming effort. So, the creation of an intelligent tool that receives one or several documents, identifies the most important information and, from that, automatically generates slide decks, can prove to be very useful since it will reduce the time and effort needed, making it possible for teachers, and trainers in general, to invest their time in other undertakings.

In order to create such a system, two subsets of Artificial Intelligence — Machine Learning (ML) and Natural Language Processing (NLP) — need to be taken into consideration.

[1] https://www.microsoft.com/microsoft-365/powerpoint
[2] https://prezi.com

NLP is needed for handling content expressed in human language, easier for humans, not so much for machines. As for ML, it can be important because, unlike algorithmic programming, it can generalize and therefore deal with cases that it has not seen before but that resemble previous data. Furthermore, while other methods summarise texts disregarding their context, ML can learn to generate from examples of human-created presentations.

1.2 Goals

The main goal of this work is to investigate machine learning and natural language processing techniques for extracting key contents from written documents, in order to summarise English or Portuguese documents, and organise the resulting summaries into slides. In this way, a first draft of a slide deck will be generated, with humans only having to review it and add or delete certain elements, reducing the time and effort required for slide development. As a result, the focus of this work is primarily on the information in the slide rather than other elements of slide decks, such as organisation, design, and visuals like images or tables. For that, various summarization methods will be examined and compared in order to determine which is most suitable for the issue at hand. Therein are methods used for slide generation that are state-of-the-art and methods that were previously only used for summarising. Also contrasted are supervised and unsupervised methods, as well as extractive and abstractive methods.

The generated slides will be subject to a quality evaluation. As follows, we can state the requirements that well constructed slides must contain. Slides should encompass all the important information, have adequate coverage of all the topics, but should not be excessive in terms of both the number of slides and the information they contain. Besides, they should have coherence of speech, meaning that from one sentence to another there must be a logical jump. Everyone should comprehend the topic of the presentation by only reading the slides. [Hashemi et al., 2012]

1.3 Proposed Approach

This section summarises the proposed approach for achieving the goals stated in the previous section.

The creation of a slide deck involves two steps: summarization and slide generation. Since the first step is the main goal of this work, several summarization methods were tested. These methods were of two types: extractive or abtractive. The extractive methods include supervised and unsupervised algorithms in order to have an understanding of which approach performs best. Some of the methods chosen were included in the state of the art for slide generation, while others were only used previously in summarization problems but showed promising results in that context. As for the abstractive methods, they are based on pre-trained transformers, so they are all supervised.

In selecting each method, consideration was given to its novelty, popularity, track record, and variety. It was made an effort to choose several methods with different approaches to summarization, i.e., there are methods based on machine learning, graphs, topics, statistics, and Latent Semantic Analysis. This allows for a variety of methods in order to understand which one is the best performing.

In order to evaluate the chosen methods, some datasets composed of pairs of documents

and slides were chosen instead of summarization-only datasets because the summaries created for slides have other requirements that other summaries do not. Having datasets allowed for an automatic evaluation of the methods. But, with this kind of evaluation, it is only possible to assess the summaries (quality and quantity of the text), not the entire slideshow. Additionally, it only contrasts the generated summary with a golden summary, even though the generated summary may differ but still represent a valuable summary. So, in an effort to lessen these limitations, a human evaluation was conducted to assess slide decks rather than just summaries, which allowed for the evaluation of other aspects of the slides, such as the organisation of the information, the overall quality of the presentation, and its viability as a starting point for a final presentation. This assessment, however, is not without its drawbacks because it can be time-consuming and highly subjective. Having both types of evaluation will allow for a broader evaluation.

1.4 Main Contributions

This thesis presented several contributions to the slide generation problem, even though its main focus was on the summarization step. A variety of experimental methods were used for this step. The methods chosen fell into two categories: extractive and abstractive. Each category had a number of methods; some of them were already mentioned in the state of the art, but most were only used in the context of summarization, not slide generation. For the extractive methods, two types were studied: supervised and unsupervised, with the latter being tested in two languages: Portuguese and English. The same happens for one of the abstractive methods where multilingual summarization is possible. Instead of summarising the entire text, another approach for the abstractive methods was also tested, which involved breaking the text up into sections and summarising each section separately. Following individual method testing, a combination of extractive methods is created in an attempt to improve the results. It relies on the redundancy of the summaries generated by different methods and selects only sentences that appear in more than one of such summaries.

The generated summaries were automatically tested using two datasets. Furthermore, through those summaries, slide presentations were created and evaluated by a group of people. These evaluations allowed us to draw conclusions about the benefits and limitations of each method. All this work will speed up the slide creation process, which is currently completely manual, in addition to identifying future directions to improve the performance of this process and further reduce the need for human intervention.

Besides the work presented in this thesis, a scientific paper [Costa et al., 2022] was written and accepted for publications in the proceedings of the XXIV International Symposium on Computers in Education (SIIE). In this paper, the unsupervised methods are tested in the slide datasets, where their performance is compared, also with that of state-of-the-art supervised methods. This comparison helps to determine whether unsupervised methods are a viable alternative because their use would make it possible to summarise every text independently of their language or topic.

1.5 Contextualization

This thesis is developed as an internship at Instituto Pedro Nunes, in the scope of the Masters in Informatics Engineering, at the University of Coimbra. The work presented

is within the scope of the project SmartEDU, which commits to creating a system that automatically generates slides and creates question-answering problems in order to assist educators by facilitating and reducing the time required for such processes. It has the participation of three organizations: Instituto Pedro Nunes (IPN), Mindflow[3], a company that uses gamification to improve and accelerate the learning processes of specific content, and the University of Coimbra through the Center of Informatics and Systems, all of which contribute resources to the proposed goal.

SmartEDU (CENTRO-01-0247-FEDER-072620) is co-financed by the European Regional Development Fund (FEDER), through Portugal 2020 (PT2020), and by the Regional Operational Programme Centro 2020.

1.6 Structure of the document

This document is divided into seven chapters, each of which contains the following information:

- Background (chapter 2): brief exposure of all the topics and terms needed to understand the rest of the thesis.
- Related Work (chapter 3): exhibits a bibliographic review regarding previous work done over the past few years towards automatic slide generation. This section contains all the different approaches used, each with an explanation of how they were implemented and their results, referencing the articles where they appear.
- Summarization Results (chapter 4): displays and describes every result of the automatic evaluation metrics for summarization in two datasets composed by scientific articles and their respective slide decks. Includes an explanation of all the methods, and datasets used, as well as the conclusions drawn from the experiences.
- Slide Generation (chapter 5): displays, for a dataset of summarization, the outcomes of the automatic evaluation when contrasting the summaries of the dataset with those produced using various summarization methods. Then, presents a few slide deck examples created from those summaries and shows their human evaluation.
- Conclusion (chapter 6): retrospective of the work, covering its main conclusions and directions for future work..

[3]https://mindflow.pt/

Chapter 2

Background

This chapter presents useful concepts for understanding the rest of the thesis. It starts with a brief explanation of what is Natural Language Processing. Then, concepts related to text preprocessing are explained, followed by a compendium of approaches used by the scientific community in different summarization methods. Finally, the most used and relevant evaluation metrics for summarization are presented.

2.1 Natural Language Processing (NLP)

Natural Language Processing is a field of study concerned with translating humans natural speech into a language that machines can better manipulate. As expected, this is not an easy task. People do not speak in a structured, clean way. In fact, they often make mistakes while speaking and writing. Grammar checkers can easily correct errors when they are simply misspelt or misused, but it is more challenging when there are other types of errors. For example, while speaking, people can mispronounce a word or have an accent that changes the sound of the word, and so machines will struggle to recognize them. However, even if humans do not introduce any mistakes while speaking/writing, it is still a challenge for machines to depict the meaning of sentences, because this is not as simple as reading a word and searching in a dictionary for its definition. Words can be written or pronounced in the same way, but have different meanings depending on the context they are in. Moreover, even if there is only one possible meaning for a word, it can have a positive or negative cognition but actually mean the opposite (irony or sarcasm). Similar to this, other figures of speech are also possible. Additionally, synonyms, i.e., different words with the same meaning, can also be a challenge, while humans may use context, machines need to know what set of words correspond to the same definition.

Furthermore, people often use different ways of speaking depending on their language or region. Words can have different meanings depending on where they are used. Additionally, every place and language has its own slang, which is constantly changing. It can be very hard to keep track of every change, especially if it is not a widely spoken language, since this is a task that requires much data and those languages do not have that. These difficulties, however, do not reduce the importance of NLP. This knowledge is used as a bridge between humans and machines and therefore accommodates a wide variety of applications, among them chatbots, virtual assistants, question answering, text prediction, language translation and text summarization.

2.2 Preprocessing

Every summarization method starts with preprocessing, which is used to get the text ready for the actual processing. It uses NLP methods to clean and manipulate data in order to reduce the text to only the words needed for the task at hand, which will enhance algorithms performance and mitigate a minority of the problems described in the previous section.

> The Carnation Revolution, also known as the 25 April, was a military coup by left-leaning military officers that overthrew the authoritarian Estado Novo regime on 25 April 1974 in Lisbon. It resulted in the Portuguese transition to democracy and the end of the Portuguese Colonial War.

Text 2.2.1: Text extracted and edited from the Wikipedia article "Carnation Revolution"

There is a set of techniques commonly used to perform this task. The text above: 2.2.1 is an edited excerpt taken from the Wikipedia article "Carnation Revolution" and will serve as an example of how the following techniques modify a text.

- Case-Folding: process of converting the given text into lower case in order to maintain the consistency flow, as it is seen in the example below:

  > the carnation revolution, also known as the 25 april, was a military coup by left-leaning military officers that overthrew the authoritarian estado novo regime on 25 april 1974 in lisbon. it resulted in the portuguese transition to democracy and the end of the portuguese colonial war.

  Text 2.2.2: Text from 2.2.1 in lowercase

- Segmentation/Tokenization: divide the text into individual segments/tokens, as it is demonstrated next:

  > 'The', 'Carnation', 'Revolution,', 'also', 'known', 'as', 'the', '25', 'April,', 'was', 'a', 'military', 'coup', 'by', 'left-leaning', 'military', 'officers', 'that', 'overthrew', 'the', 'authoritarian', 'Estado', 'Novo', 'regime', 'on', '25', 'April', '1974', 'in', 'Lisbon.', 'It', 'resulted', 'in', 'the', 'Portuguese', 'transition', 'to', 'democracy', 'and', 'the', 'end', 'of', 'the', 'Portuguese', 'Colonial', 'War.'

  Text 2.2.3: Text from 2.2.1 tokenized

- Stop word removal: remove the most common words of the language used on the respective text, which generally include determiners, pronouns and prepositions. By doing this, those words will not have any impact on the text processing, which can have a lot of advantages. If for example, the objective of a function is to select the most important words only taking into consideration their frequency, these words would be the top ones, while in fact they are not relevant. So, because they do not add any important information to the text they are removed. Below is the example text (2.2.1) stripped of all stop words.

  > The Carnation Revolution, known 25 April, military coup left-leaning military officers overthrew authoritarian Estado Novo regime 25 April 1974 Lisbon. It resulted Portuguese transition democracy Portuguese Colonial War.

  Text 2.2.4: Text from 2.2.1 striped of stop words

- Stemming: Process of reducing a word to its root. The text example (2.2.1) with this technique has the following aspect:

> The Carnat Revolut , also known as the 25 April , wa a militari coup by left - lean militari offic that overthrew the authoritarian Estado Novo regim on 25 April 1974 in Lisbon . It result in the Portugues transit to democraci and the end of the Portugues Coloni War .

Text 2.2.5: Stemming of the text from 2.2.1

- POS Tagging: For each word identify its corresponding part of speech, meaning that each word word will have a tag that identifies if it is a noun, verb, adjective, adverb, preposition, conjunction, pronoun or interjection, among others. This tag will be determined based on the word definition and context. The images below demonstrate how this is done:

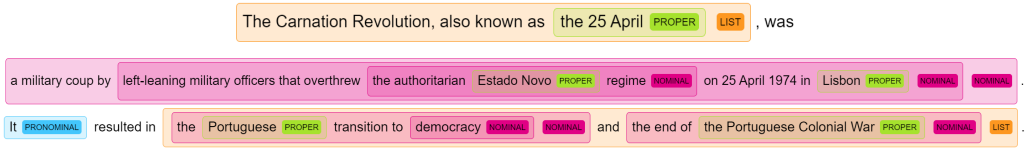


Figure 2.1: Pos Tagging of an excerpt based on the Wikipedia article "Carnation Revolution"

- Chunking: instead of only handling individual words, chunking extracts phrases from text, that could be important together. This is done by grouping some words with a specific tag given by the output of POS Tagging. The chunks obtained from the text example (2.2.1) are:

> The/DT Carnation/NNP Revolution/NNP; the/DT; April/NNP; a/DT; coup/NN; the/DT; Estado/NNP Novo/NNP regime/NN; April/NNP; Lisbon/NNP; the/DT; transition/NN; democracy/NN; the/DT end/NN; the/DT; Colonial/NNP War/NNP;

Text 2.2.6: Chunks obtained from 2.2.1

- Coreference Resolution: finds all the words in a text that make reference to another word and replaces those words by the one it mentions. For example, in the second sentence of the text displayed in 2.2.1, the word "It" would be replaced with "The Carnation Revolution".

- Dependency Parsing: analyses the grammatical structure of a sentence and creates links between a head word and some other word representing the existing dependency between them. For each link there is a respective tag identifying the type of dependency. There are 37 types of existing dependencies, such as coordinating conjunction, compound, conjunct, determiner, nominal subject, root, etc. A very popular dependency parser is the Stanford Parser [Chen and Manning, 2014]. For example, for the second sentence in the text displayed in 2.2.1 the dependencies found are those shown in figure 2.2.

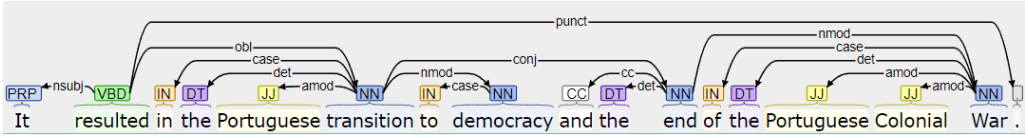


Figure 2.2: Dependency Parsing of an excerpt based on the Wikipedia article "Carnation Revolution"

2.3 Summarization

Summarization is the act of shortening a text into only its main ideas. In order to do this automatically, it can be tackled from two angles: extractive and abstractive. Extractive simply consists of selecting the sentences that best describe a document as a whole. For this, three steps are taken: preprocessing, which was already mentioned, sentence scoring and sentence selection.

Sentence scoring is the most laborious and complex step. It involves giving a score to each sentence based on its relevance in the text. Possible methods can be found in chapter 3, with all of the concepts required for comprehension in the subsections that follow.

Sentence Selection involves selecting the best sentences of the candidates obtained in the previous step. This is done usually in two ways: greedy or through Integer Linear Programming (ILP). Greedy is simply selecting the N sentences higher in the ranking. ILP is a type of optimization problem where the variables are integer values and the objective function and equations are linear. The objective function involves a maximization or minimization of one or several variables and has a certain number of constraints associated with it.

As for abstractive summarization, it is a closer approach to human text representations, because it creates novel sentences, either by rephrasing or using other words. So, abstractive presents more readable, concise and cohesive summaries. Furthermore, these summaries may be smaller, because abstractive only keeps the important parts of a sentence, while extractive uses whole sentences. However, as expected, abstractive summaries are harder to produce. Therefore, we can find many studies that use extractive methods.

Below is an excerpt of the Wikipedia article "Coimbra" and its corresponding extractive and abstractive summaries.

Coimbra is a city and a municipality in Portugal. This was in large part helped by the establishment of the University of Coimbra in 1290, the oldest academic institution in the Portuguese-speaking world. Its historical buildings were classified as a World Heritage site by UNESCO in 2013: "Coimbra offers an outstanding example of an integrated university city with a specific urban typology as well as its own ceremonial and cultural traditions that have been kept alive through the ages."

Text 2.3.1: Extractive summary of the text in 2.3.3

Coimbra is the second-largest urban area in Portugal outside Lisbon and Porto Metropolitan Areas. Its historical buildings were classified as a World Heritage site by UNESCO in 2013. About 460,000 people live in the Região de Coimbra, an area of 4,336 square kilometres (1,674 sq mi).

Text 2.3.2: Abstractive summary of the text in 2.3.3

Coimbra is a city and a municipality in Portugal. The population of the municipality at the 2011 census was 143,397, in an area of 319.40 square kilometres (123.3 sq mi). The second-largest urban area in Portugal outside Lisbon and Porto Metropolitan Areas after Braga, it is the largest city of the district of Coimbra and the Centro Region. About 460,000 people live in the Região de Coimbra, comprising 19 municipalities and extending into an area of 4,336 square kilometres (1,674 sq mi).
Among the many archaeological structures dating back to the Roman era, when Coimbra was the settlement of Aeminium, are its well-preserved aqueduct and cryptoporticus. Similarly, buildings from the period when Coimbra was the capital of Portugal (from 1131 to 1255) still remain. During the late Middle Ages, with its decline as the political centre of the Kingdom of Portugal, Coimbra began to evolve into a major cultural centre. This was in large part helped by the establishment of the University of Coimbra in 1290, the oldest academic institution in the Portuguese-speaking world. Apart from attracting many European and international students, the university is visited by many tourists for its monuments and history. Its historical buildings were classified as a World Heritage site by UNESCO in 2013: "Coimbra offers an outstanding example of an integrated university city with a specific urban typology as well as its own ceremonial and cultural traditions that have been kept alive through the ages."

Text 2.3.3: Excerpt adapted from the Wikipedia article "Coimbra"

In the next sections are explained several concepts that can be used for abstractive and extractive summarization.

2.3.1 Statistical summarization Methods

This section presents several statistical methods used to calculate the similarity between words/sentences. These are statistical because they only use mathematical formulas, not taking into account any linguistic properties of the text. They treat words like mathematical events and do not worry about their meaning or context.

TF-IDF [Luhn [1958] and Jones [1972]] evaluates the importance of a word in a document taking into account several other documents. First, the frequency of a word (w) in a document (d) is calculated, by the equation 2.1.

$$TF = log(count(w, d) + 1) \tag{2.1}$$

Logarithm is used to attenuate the high frequency of common words.

Then, the Inverse document frequency is calculated by the equation 2.2.

$$IDf = log(\frac{N}{dft}) \tag{2.2}$$

where N is the total number of documents and dft is the number of documents in which word w occurs.

So, the final importance of a word is given by:

$$TF.IDF = TF \times IDF \tag{2.3}$$

Jaccard is another statistical method used to calculate the similarity between two sets. Taking A and B as sets, the Jaccard similarity is calculated as follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.4}$$

So, for example, for A = {1, 2, 3, 4} and B = {3, 4, 5}:

$$\begin{gathered} \text{A} \cap B = \{3, 4\} = 2 \\ \text{A} \cup B = \{1, 2, 3, 4, 5\} = 5 \\ \text{J(A, B)} = \frac{2}{5} \end{gathered}$$

2.3.2 Graph summarization Methods

A graph is a structure represented by a pair of nodes and edges, where the edges connect the nodes that are in some way related. In summarization, a type of graph called a "semantic network" is used, where the nodes correspond to sentences and are connected by links (edges) of different sizes, depending on how related the two connected sentences are, i.e., the shorter the link, the more related the sentences.

In order to determine the relevance of the nodes in this network, spreading activation [Collins and Loftus, 1975] is used. This algorithm starts by defining the value of all the nodes as zero, except for one or more nodes that will be the origin and have a value of one. Then each of the following nodes receives a value, taking into account a certain weight and a decaying factor. This factor is used because the further the tree is searched, the weaker the relatedness of the nodes with the origin, and therefore its activation values should be smaller than the ones in the beginning. Figure 2.3 shows an example of this algorithm.

Ontologies are one type of graph among many others. Within a knowledge domain, an ontology is a graph that represents a set of concepts and their relationships. Because ontologies are a type of network with multidirectional connections between nodes, they are used instead of trees to represent more than a hierarchical structure.

The concepts that the ontology is built upon can also be referred to as classes. For example, a class can be a programming language, with instances concerning the several existent languages like Java, Python, C, etc. Instances and classes can have a set of properties that describe them. Furthermore, each class can have subclasses restricting the respective instances. Ontologies are composed of a set of concepts and their respective instances form a knowledge base.

In order to define and instantiate ontologies on the Web, the World Wide Web Consortium (W3C) created the Web Ontology Language (OWL) [McGuinness, 2004]. This language allows the description of concepts and categories and the relationships between them in documents and web applications.

In addition to ontologies there are other ways to link words/sentences. An example of this is WordNet [Fellbaum, 1998]. This is a lexical database of semantic relations between words, where words are linked by semantic relations, which means that if there is a relation between the meanings of two words, they become linked in WordNet. It is used for semantic analyses or as a dictionaire where it can be found sets of synonyms called synsets. Each synset can have a set of hypernyms, hyponyms, holonymys, and meronyms. Thus, for every word, there are connections with these sets of synonyms, hypernyms, etc., in a tree-like hierarchical structure. As such, WordNet finds its primary use in automatic text analysis

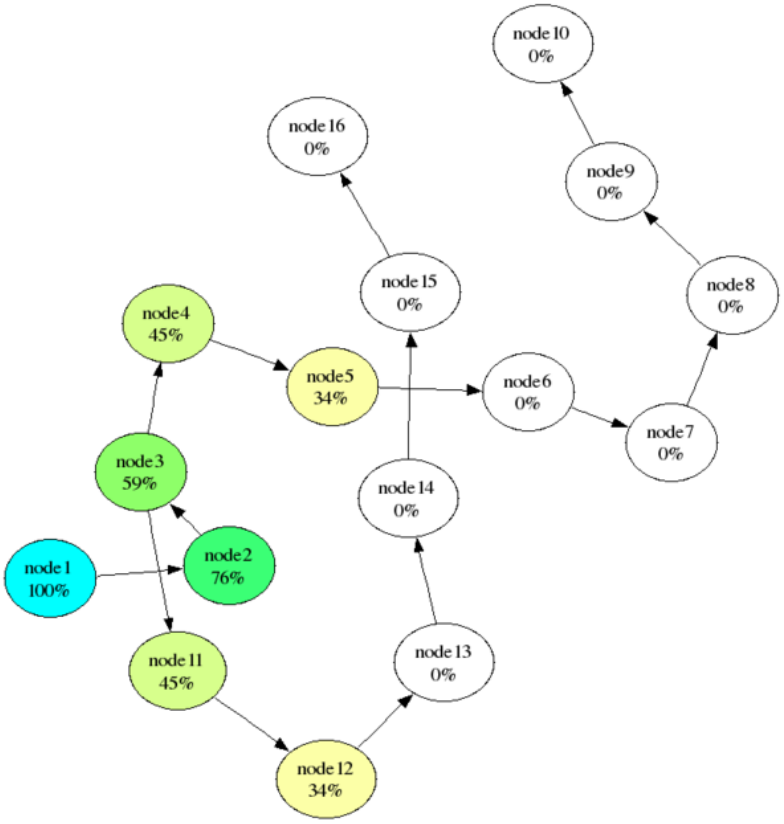


Figure 2.3: Example of spreading activation originated at node 1, with a weight of 0.9, for every link, and a decay factor of 0.85. Taken from Reed [2008]

and artificial intelligence applications, as it is a powerful tool to examine the relationships between words in text. For summarization this can be usefull since it allows us to group words by their meanings/synsets, resulting in the identification of the several topics that exist on a document. Having topics will help us determine which parts from the text should be included in the summary.

2.3.3 Machine Learning

Machine Learning (ML) is a field of Artificial Intelligence (AI) that aims to study several algorithms that do not need to be explicitly programmed. Instead, much like humans, they learn through experience and data, gradually improving the results based on what they learned in the previous interaction. One of the most common uses of ML is for classification problems. It makes predictions about a piece of data's label using prior data and their respective labels. ML can resort to supervised or unsupervised learning. Supervised learning uses labelled datasets, that is, data that already has the correct classification assigned. It usually resorts to methods such as neural networks, linear regression, random forest, support vector machine (SVM), etc. Unsupervised learning uses unlabeled data, so the algorithms need to find information on their own by scanning the data for patterns. That allows for more complex tasks with bigger datasets. It usually resorts to methods such as neural networks, probabilistic clustering methods, etc.

In order to use ML for summarization, words need to be transformed into vectors because ML methods cannot process words as input, only numerical data. So, words are transformed into word embeddings, that is, a vector representation of the meaning of a

word, such that words with similar meanings will have similar vector representations. One way to generate these embeddings is through the Word2Vec method, which can be learned by two methods: the Continuous Bag-of-Words and the Continuous Skip-Gram Model. The first one learns the embedding by predicting the current word based on its context, and the second one does the opposite: it predicts the surrounding words based on the current one.

In the state of the art for automatic generation of slides (see chapter 3), three different machine learning methods are explored: random forest, support vector regression, and neural networks. These methods are further explained below.

Random forest (RF) follows three steps. Firstly, it selects random samples from a dataset and constructs a decision tree for each sample, such that the final prediction is the one that was chosen more times by the trees. A decision tree maps the possible outcomes of a series of related choices and is composed of three modules: root, node, and leaf. The roots are the initial sample, then the sample is divided into several branches until reaching the leafs, where it stops. The nodes represent the several attributes that will be used to calculate the prediction.

Support Vector Machines (SVM) aim to find a line that separates the data into classes, in another words a hyperplane, defined by:

$$w^T x - b = 0 \tag{2.5}$$

where w are the coefficients, and x the predictors (feature).

However there can be more than one line that separates the classes. In order to choose one of the lines two hyperplanes that separate the two classes of data are selected, so that the distance between them is as large as possible. The points closest to the lines from both classes are called support points. These hyperplanes can be described by the following equations: $w^T x - b = -1$ and $w^T x - b = 1$

Then the distance between the support points and the line (margin) is computed. The hyperplane with the greatest margin is the best hyperplane. Geometrically, the distance between these two hyperplanes is $\frac{2}{||w||}$ so to maximize the distance between the planes $||w||$ must be minimized. However, this only works for data that can be separated linearly. For non-linear data some patterns are allowed to be in the wrong margin but they will suffer a penalization (slack variables). For that there is the following optimization term:

$$\begin{aligned} min \quad & \frac{1}{2} w^T w + C \sum \xi_i \\ s.t. \quad & y_i(x_i^T + b) \geq 1 - \xi_i \xi_i \geq 1 \end{aligned} \tag{2.6}$$

In the equation above, y are the targets and ξ is the deviation from the margin. C is a free parameter that influences the margin, with a greater C the margins are smaller, giving a greater penalization to the bad classifications, and with a lower C the margins are higher, not penalizing much the wrong classifications.

Support Vector Regression (SVR) uses the same principle as SVM, but for regression problems. SVR allows for defining a threshold for the maximum error (ϵ), also taking into consideration the deviations from the margin that should be as little as possible. So, the objective function and constraints are similar to equation 2.6 but with this additional parameter:

$$\begin{aligned} min \quad & \frac{1}{2}||w||^2 + C\sum|\xi_i| \\ s.t. \quad & |y_i - w_i x_i| \leq \epsilon + |\xi_i| \end{aligned} \tag{2.7}$$

Artificial Neural Networks (ANN) are networks that seek to imitate how human brains work, particularly the way that neurons signal to each other. ANNs are composed of three node layers: input, hidden layers, and output. Multilayer Perceptrons (MLP) refer to networks that connect multiple layers in a directed graph. In ANNs, the nodes are connected with a certain weight and threshold. The threshold determines which nodes should be activated. If a node's value, obtained by an activation function, is greater than the threshold, the node is activated, and the data is transmitted to the next layer; otherwise, the data is ignored. Activation functions are used in order to introduce non-linearity to the model; otherwise, even though the system would be simpler, more complex operations would not be possible.

Among these networks, there are two architectures that are used for the automatic generation of presentation slides (see chapter 3): Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN).

Convolutional Neural Networks [LeCun et al., 1998] are most often used to solve classification and image problems and are composed of three layers: Convolutional, Pooling and Fully-connected (FC). Figure 2.4 shows a representation of this network.

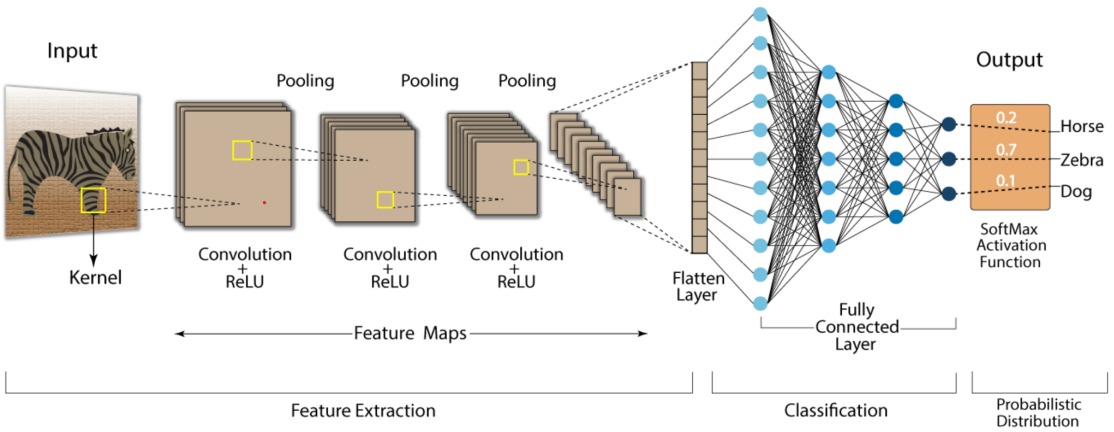


Figure 2.4: Visual representation of the CNN network. Image taken from Swapna. [2022]

The convolutional layer performs a convolution process by using the input data and the filter that it receives. This process is done for every filter and consists of the use of a filter/feature detector that will move across the input matrix in order to determine if a given feature is present. This detector starts at a certain area of the input and calculates the dot product between the pixels of both the input and the filter, then moves a given number of pixels (stride) to the next area of the input and calculates the respective dot product, and so on until it reaches the end of the input. The final result will be a junction of all the dot products, known as a feature map. A Rectified Linear Unit (ReLU) is applied to the feature map after each convolution: (max(0, x)).This process is done for every filter, generating a map for each of them that will later be joined together to form the output. The output can have different sizes that will change accordingly to the number of filters, the stride and the padding. Padding can be valid, equal, or full.Valid is equal to 0 padding, meaning that the output is going to be the size of the filter. Same refers to a padding that

makes the size of the output equal to the size of the input, and Full increases the size by adding zeros to the border of the input.

There can be more than one convolutional layer, constructing a hierarchical architecture where the first layer corresponds to the Low-Level features. For example, in an image, it would be the edges, color, and gradient orientation, and the following layers would combine the previous information to generate higher level features. Combining certain edges and colors would give a certain part of the image. Those parts could later be combined to form the whole image.

The pooling layer is responsible for reducing the dimensionality of the data, recurring to a very similar approach to the previous layer, also using a filter. For this, there are two types of pooling, max and average. Max is the one that, generally, presents the best results and consists of extracting the maximum value from the part of the input covered by the filter. Average extracts the average of each part of the input covered by the filter.

Fully-connected (FC) layer connects all the nodes in one layer to the nodes in the other layer. This is applied to classification tasks using a softmax activation function that will classify the inputs with a probability from 0 to 1.

CNNs and other feed forward neural networks are meant for independent data points, so when applied to a NLP problem that seeks to depict the next word in a text, these networks would only have the current word as data. However, as mentioned in section 2.1, this would not be ideal, because there would not be any context for the word. So, in order to solve this problem, a Recurrent Neural Network (RNN) could be used. These networks treat data like a sequence, with the current data depending on the previous data, which means that, in the problem stated before, we would have access to the previous words, therefore having context, which would allow the prediction of the next word with more accuracy. This is possible because RNNs keep an internal memory that stores the data of the previous inputs in order to generate the next output (hidden state).

This network works like any other neural network. It is composed of the usual layers: input, hidden layers, and output. Each cell contains weights and a respective activation function, but in order to keep a memory, contrary to other networks, each cell receives two inputs: the current word vector and the hidden state. The hidden state is going to be added, with a certain weight, to the present word vector. The activation function is then applied to each output and passed to the next cell, with the following functions being used in this case: Sigmoid, Tanh, and Relu.

However, there is a problem with this because, with the passing of time, the oldest data gets forgotten among new data, weights, and activation functions, so this only works for short-term memory. Variations of RNNs, such as LSTM and GRU, are used for long-term memory.

The difference between Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber [1997], Phi [2020]] and RNNs resides in the interior of the cells. Instead of having only the current data and the hidden state combined and passed through an activation function to form the output, LSTM cells have three gates that regulate all the information. Each cell calculates the hidden state and the cell state that will be passed to the next cell. This last one is used to keep track of the previous data. Figure 2.5 shows a representation of a cell of this network.

For the cell state, the forget and input gates are utilized:

- Forget gate: determines which previous data to discard or keep. For that, the current

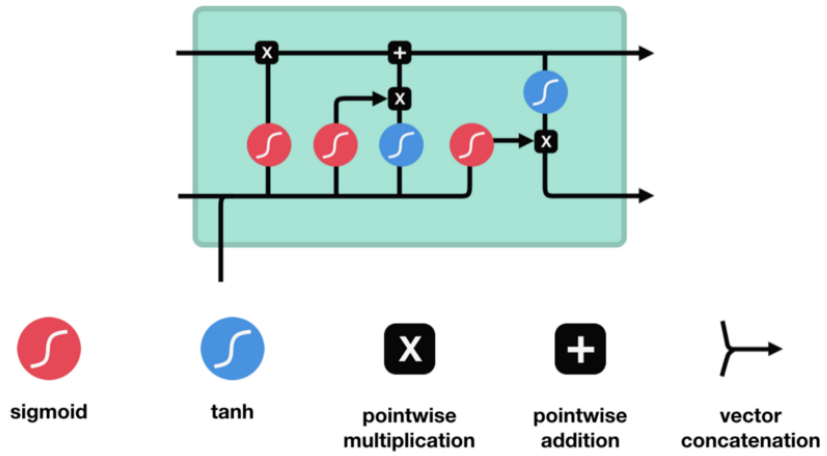


Figure 2.5: A visual representation of a cell in the LSTM network. Image taken from Phi [2020]

word vector and the hidden state are passed through a sigmoid function that returns values between 0 (discard) and 1 (keep). The ones closest to 0 are less important, and the ones closest to 1 are the most important.

- Input gate: determines which new data to keep. The current word vector and the hidden state go through a sigmoid and a tanh function in parallel. The sigmoid returns the values between 0 and 1, deciding in this way which ones to update. The tanh transforms the data into values between -1 and 1. Then, the output of both functions is multiplied, resulting in the values given by the tanh filtered by the sigmoid.

The outputs of the forget and input data are used to determine the cell state. For that, the previous cell state is element-wise multiplied by the forget gate values, and then the output of that operation is element-wise added to the input gate values.

After having the cell state, the hidden state is calculated using the output gate. A multiplication of a tanh function, which receives the cell state, and a sigmoid function, that receives the current word vector and the hidden state, is used to achieve this.

LSTM has a variation named "Bidirectional Long Short-Term Memory (BiLSTM)", where instead of only one LSTM that takes the input forward, there is another one that takes it backwards, allowing us to have not only the previous words but also the following ones, considering, in this way, even more context. Figure 2.6 shows a visual representation of this network.

The other option for long term memory is the Gated Recurrent Unit [Kostadinov [2019], Cho et al. [2014]]. GRU is similar to an LSTM, but instead of using the cell state and the hidden state separately, they are combined together in the hidden state. Furthermore, in GRU, there are only two gates. Figure 2.7 shows a representation of a cell of this network.

- Update gate: like the forget and input gates, it determines which data to discard or keep. For that, the current word vector and the hidden state are passed through a sigmoid function.

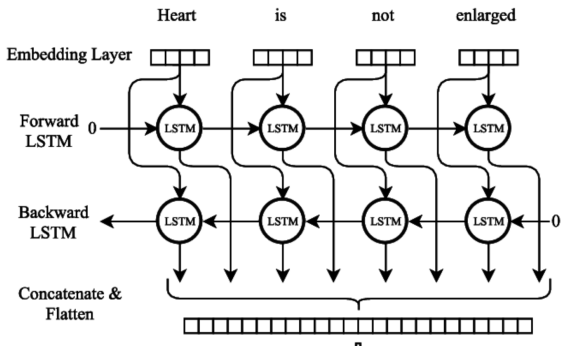


Figure 2.6: Visual representation of the BiLSTM network. Image taken from Cornegruta et al. [2016]

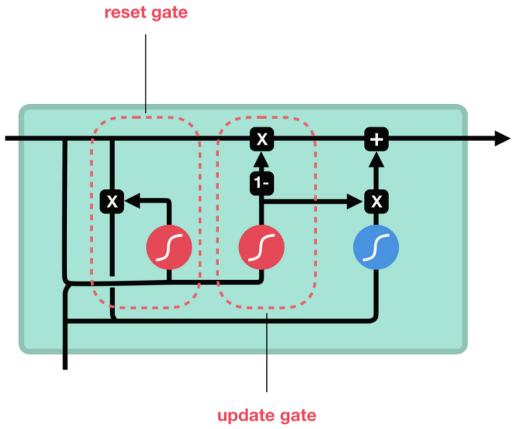


Figure 2.7: A visual representation of a cell in the GRU network. Image taken from Phi [2020]

- Reset gate: determines how much past data to discard. It is calculated the same way as the previous gate; the only things that change are the weights attributed to the current word vector and the hidden state.

After having both gate outputs, they are used to calculate the next hidden state. Firstly, a tanh function is calculated with the sum of the current word vector and the value obtained by element-wise multiplying the reset gate with the hidden state as inputs. Then, the output of that function is element-wise multiplied by $1 - output\ of\ the\ update\ gate$ and added to the value obtained by the element-wise multiplication of the update gate with the hidden state, in order to obtain the final result corresponding to the next hidden state.

Such as LSTM that has a variation that creates two LSTMs one for taking the input forward and other for taking it backwards, GRU also has a variation called Bidirectional Gated Recurrent Unit (biGRU) that creates two GRUs with the same purpose.

In addition to the previous networks that were used for slide generation, there are other architectural approaches, such as transformers, that can be used for the summarization problem within slide generation. Figure 2.8 presents the transformers architecture.

Transformers have an encoder-decoder structure. The encoder (left side of figure 2.8) starts by receiving words and transforming them into vector embeddings. Then, each vector is augmented. This happens by summing to the embedding vector a positional encoding

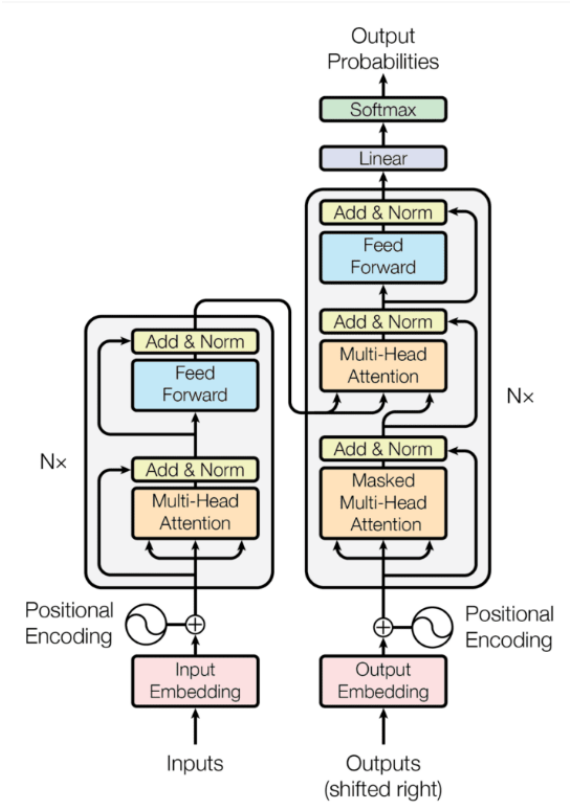


Figure 2.8: The Encoder-Decoder Structure of the Transformer Architecture. Image taken from Vaswani et al. [2017]

vector. After that, the resulting vector is passed to the Multi-Head Attention Layer. This layer receives as input a query that represents the current state, position or time step of the network; a value, i.e., things the network is going to pay attention to; and a key that is used to determine how much attention is paid to its corresponding value. Then, those variables are used to produce an encoded representation with attention scores, i.e., weights, for each word that is given in the input. This calculation is repeated in parallel several times and is then combined to produce the final attention score. After that, the encoding is passed to the fully-connected (FC) layer (explained above in the CNN network).

As for the decoder (right side of figure 2.8) it receives as input the last output at time step t-1. The input is then augmented by positional encoding, such as in the encoder. The resulting vector is then passed through a masked multi-head attention layer, that is very similar to the first layer of the encoder but has a mask over the values produced by the multiplication of the query and key. After that, the output is passed through a multi-head attention layer and a FC feed-forward network layer, as in the encoder. The output of the decoder is then passed on to an FC layer and a softmax layer, which use it to predict the next word in the output sequence.

Artificial Neural Networks are currently the most popular methods for machine learning. However, they are not perfect and may not be ideal on some occasions. In fact, ANNs require a large amount of input data, and even though that may help with complex applications, it makes the process slow. So, for more trivial problems, other methods like Support Vector Machines and Random Forest may be preferable since they require less input and data preprocessing, and therefore less runtime.

2.4 Evaluation Metrics

This section is on evaluation metrics commonly used for determining the quality of automatically generated text, including summaries or slide decks. Particularly, these include automatic measures like ROUGE [Lin, 2004], BERTSCORE [Zhang et al., 2020], BLEURT [Sellam et al., 2020] and manual evaluation, by humans.

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. As input, it receives the automatically-generated summary and the respective gold summary, and compares them returning several different metrics, each with its corresponding recall, precision and f-score. These metrics are:

- ROUGE-N measures the overlap of sequences of N words (i.e., n-grams) between the hypothesis and the reference summaries.
- ROUGE-L measures the longest common subsequence (LCS) between the hypothesis and the reference summaries. In other words, the longest sequence of words shared by both summaries.
- ROUGE-W measures the weighted LCS-based statistics that favors consecutive LCSes between the hypothesis and reference summaries. It is similar to ROUGE-L, but it also tracks the lengths of consecutive matches, in addition to the length of the LCS.
- ROUGE-S measures consecutive words from the reference summary that appear in the hypothesis but are separated by a defined number of other words (i.e., skip-grams).
- ROUGE-SU is a combination of ROUGE-S and ROUGE-1.

The recall metric is calculated by dividing the number of matches between the generated and the gold summaries by the length of the gold summary. The precision metric is calculated by dividing the number of matches between the generated and the gold summaries, by the length of the generated summary. It is important to remember that the matches to be counted in the numerator depend on the ROUGE metric being used. For example, for ROUGE-N it is the number of matching n-grams while for ROUGE-S it is the number of matching skip-grams. The F-score is given by the harmonic mean of the precision and the recall:

$$F1 = 2\frac{Precision * Recall}{Precision + Recall} \tag{2.8}$$

These metrics are only used to evaluate text summarization tasks. So, to evaluate a presentation generated automatically, it can take the text in the slides and evaluate it as a simple summarization problem, but all the other aspects specific to the task are not taken into account. With this in mind, Fu et al. [2021] created several metrics that take inspiration from ROUGE, in order to answer this problem. These new metrics evaluate not only the text, but also the included figures in the slides and their respective layout:

- Slide-Level (SL) ROUGE: takes into account the number of existent slides, because the presentation should not be too long and each slide should not have a lot of information.

$$ROUGE - SL = ROUGE - L \times e^{\frac{|Q-\tilde{Q}|}{Q}} \tag{2.9}$$

Q and $\tilde{Q}$ represent the number of slides in the generated slides decks and in the gold standard slide decks, respectively.

- Longest Common Figure Subsequence (LC-FS): applies ROUGE-L but instead of text compares the list of figures in the generated slides to the gold ones.
- Text-Figure Relevance (TFR): evaluates how well correlated the text and figures truly are.

$$TFR = \frac{1}{M_F^{in}} \sum_{i=0}^{M_F^{in}-1} ROUGE - L(S_i, \tilde{S}_i) \tag{2.10}$$

in represents the input. S_i and $\tilde{S}_i$ are sentences from the generated and gold slides, respectively, that contain images for the input corresponding to i.

- Mean Intersection over Union (mIoU): evaluates the layout of the slides. This is similar to the metric stated before, however, because it refers to the layout instead of ROUGE-L, uses IOU, which calculates the overlap between areas and not words.

$$mIoU(D, \tilde{D}) = \frac{1}{N_O^{out}} \sum_{i=0}^{N_O^{out}-1} IoU(D_i, \tilde{D}_{J_i}) \tag{2.11}$$

D and $\tilde{D}$ represent the generates slide deck and gold standard slide decks, respectively.

J_i belongs to a list that achieves the maximum mIoU between slide decks in an increasing order, in order to prevent mismatches between comparisons.

ROUGE is based on word overlap and thus has limitations when, despite transmitting the same meaning, the generated summaries paraphrase the gold ones using different words (e.g., synonyms). This is common in abstractive summarization. BERTScore [Zhang et al., 2020] is an alternative automatic evaluation metric for text generation which, instead of words, consider a representation of the meaning of the text. For this, it exploits contextual embeddings obtained from a BERT [Devlin et al., 2019] neural language model. Given a reference and a candidate text, contextual embeddings are obtained for each token. These embeddings are influenced by the context, meaning that, for different contexts, the same token will have different embeddings. The cosine similarity is then computed between each pair of contextual embeddings from the tokens of each summary and the most similar tokens from each summary are paired for this measure. With the similarity computed, the last step of the metric is to calculate the recall, precision and F1 score, as follows:

$$Recall = \frac{1}{|r|} \sum_{r_i \in r} \max_{c_j \in c} r_i^T c_j \tag{2.12}$$

$$Precision = \frac{1}{|c|} \sum_{c_j \in c} \max_{r_i \in r} r_i^T c_j \tag{2.13}$$

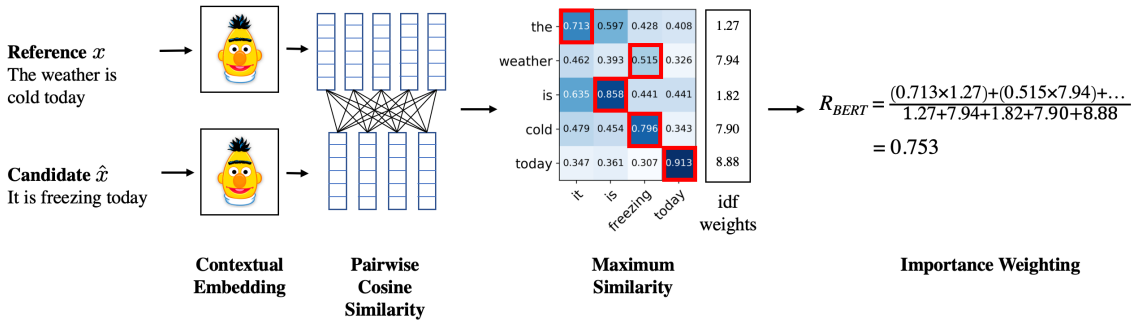


Figure 2.9: Visual representation of the calculation of recall in the BERTSCORE system, taken from Zhang et al. [2020]

where r is a reference and c a candidate. F1-score is computed as in ROUGE. Figure 2.9 is a visual representation of how recall is computed with BERTScore.

This metric allows also for the (optional) implementation of different importance weights to each word. For example, rarer words may give a better indicator of sentence similarity than common words, and for that reason it can be of interest to have different weights for each. For that, the inverse frequency of the document (idf) is calculated for each token:

$$idf(w) = -log\frac{1}{M}\sum_{M}^{i=1} I[w \in x^{(i)}] \tag{2.14}$$

where M is the number of reference sentences and I is an indicator function.

As a similar alternative to BERTScore there is BLEURT [Sellam et al., 2020], a machine learning-based automatic metric that can detect many similarities between sentences. It is composed of three main steps. Firstly, it applies BERT [Devlin et al., 2019] pre-training, which in our tests was done through the RemBERT model [Chung et al., 2020], then applies pre-training on synthetic sentence pairs, and finally, it is fine-tuned on a public collection of ratings (the WMT Metrics Shared Task dataset) as well as the user's own additional ratings. Optionally, it can still apply fine-tuning on application-specific human ratings in order to have better control of the domain utilized.

The previous methods are all automatic, meaning that they are employed through the use of algorithms based on mathematical models to evaluate how good a summary is. However, this may not be completely reliable, since summarization is a task that works with language and not mathematics and there are linguistic properties that an algorithm cannot process. For that, the only alternative is evaluation by humans. In this case, instead of an algorithm evaluating the task, humans can assess not only the summarization but, especially, the generation of slides.

To accomplish this kind of evaluation, texts and respective outputs (summaries or slide decks) are shown to a group of people that rate them accordingly to predefined criteria [Ermakova et al., 2019, Fabbri et al., 2021, Gupta and Gupta, 2019, Steinberger and Jezek, 2009, Sun et al., 2021a], such as:

- Readability: the slides are coherent, concise, fluent, and grammatically correct.
- Informativeness: the slides provide a good amount of information, covering the most important contents of the text.

- Consistency: the generated slides are similar to the gold ones.

These criteria can be presented as questions that humans need rate. For example, Sravanthi et al. [2009] chose eight papers, generated slides from them, and set some questions related to the readability and informativeness of the slides decks:

- How much information is covered in the presentation?
- What is the level of coherence in the slides?
- How much do you think this presentation could be a good starting point to prepare for the final presentation?
- What is your overall satisfaction level with the presentation?

These questions were asked to the authors of the respective papers.

Fu et al. [2021] provided the slide decks generated by their approach (Deck B) and the gold-standard slide decks (Deck A) to a group of people, and ask a few questions related to their consistency, such as:

- Looking only at the TEXT on the slides, how similar is the content on the slides in DECK A to the content on the slides in DECK B?
- How well do the figure(s)/tables(s) in DECK A match the text or figures/tables in DECK B?
- How well do the figure(s)/table(s) in DECK A match the TEXT in DECK B?

However, human evaluation has also some issues. Firstly, it is subjective, since everyone has different experiences and opinions. For example, if the evaluation task is given to people in the area of expertise of the content in the papers and to people expert in linguistics, it is normal that they have different ratings in the readability and informativeness criteria. Even if people come with the same background, their opinions will not be the same. So, it is crucial to have this in mind when choosing the judges for the task [Iskender et al., 2021]. Additionally, it is important to have a good number of people rating the outputs of the system. The more people involved in an evaluation, the more one can trust in its results. The downside is that this can rebound and generate a high variance of ratings [Belz and Kow, 2010], making it hard to take conclusions. Furthermore, contrarily to the previous evaluation metrics, human evaluation is not automatic. A set of judges need to be chosen, they need to read the original document or documents and the slides decks or summaries, and evaluate them, which can be too laborious and take much time. Not to mention that forcing all the participants to read all the documents can be impractical and hardly controllable.

2.5 Conclusion

This chapter is used as an introduction of several concepts and algorithms that are going to be used later. It begins by briefly outlining natural language processing and its significance. The various steps of summarization are then presented, beginning with preprocessing, which is used to prepare and clean the data for the actual processing. After

that, statistical, graph, and ML summarization methods are presented, with both extractive and abstractive summarization being included. Within ML, several methods are described: Random Forest, SVR, RNN, CNN, LSTM, BiLSTM, GRU, biGRU, and transformers. The metrics used to evaluate the summarization methods are then shown. Automatic metrics like ROUGE, BERTScore, and BLEURT are among them, as are human evaluation methods.

The concepts covered and examples provided in this section will be used in the research described in the following section.

Chapter 3

Related Work

This chapter presents the most relevant scientific research, in the context of automatic generation of presentation slides, from the last years.

In order to create slide decks, most related work follows two main steps, as depicted in figure 3.1: summarization of the most important topics in the text, and placement of the summary in slides.

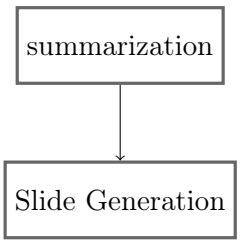


Figure 3.1: Pipeline of the generation of slides

summarization is the first and most studied step, since the main objective is not the aesthetics of the slide but its content, and so having the best possible summary of the original content is crucial. Having the summary, the only step left involves finding the best way to generate slides, which can also be complicated, but it is not the main issue studied in this report. With that in mind, this chapter is divided into two sections representing the two methods to summarization: extractive and abstractive, which are applied to the generation of slides.

For each section, several papers on slide generation are reviewed and their main contributions are highlighted, identifying the different methods, evaluation metrics, and resources used, among other features that may be of importance.

3.1 Extractive methods

Extractive methods are simpler [Nenkova and McKeown, 2012], because they only involve extracting the most important sentences of a given text, without the need of changing any word. summarization in extractive methods can be divided into two sub-steps: sentence scoring and sentence selection, meaning that the sentences must be scored accordingly to their importance and selected according to given criteria. Among these sub-steps, Sentence Scoring is the most diverse and complex. It is where the candidate sentences to use on

the summarization are isolated from the rest, so it is important that those are the ones that better encapsulate what is transmitted in the original text. For this, there are several different methods. The following subsections describe some of the extractive methods to summarization that have been applied in the automatic generation of presentation slides.

3.1.1 Statistical Methods

The most straightforward approach to this problem is to use statistical methods that determine which sentences are more likely to be important.

Christian et al. [2016b], Sariki et al. [2014] presented a very standard technique. It starts by applying preprocessing, such as: case-folding, tokenization, stop word removal, and stemming. After that, it analyses the remaining words in every sentence and gives them a score based on the following characteristics: word frequencies, similarity with the document title, location in text, and cue-phrase, that is, words that impact the importance of a sentence.

Christian et al. [2016a] studied a similar approach where its applied preprocessing, such as the method above, and the TF-IDF (explained in section 2.3.1) is calculated for every word in the text. The score for each sentence is then determined by adding the TF-IDFs of all the words in that sentence. The sentences with the highest scores are chosen for the summary.

These are very simple and fast-to-implement systems, yet they can fall short in quality. This is because it is only based on mathematical concepts and not on semantic or linguistic knowledge.

3.1.2 Discourse Based

In contrast to the last method, the discourse-based method does not rely exclusively on mathematical methods. Instead, it considers the semantic structure of a document to understand the importance of sentences in the text. Utiyama and Hasida [1999] used the GDA tagset [Nagao and Hasida, 1998] to semantically annotate documents. This annotation has three components: a parse-tree that annotates the syntactic structure; a semantic relation that encodes a relationship between an element and the element that syntactically depends on it; and a coreference that detects coreferences (i.e., different expressions that refer to the same thing), with an id for each. With this additional information from the added annotations, the system determines the topics in the document by searching for syntactic subjects that will be classified according to their referents, i.e, a class will be created for each subjects with their referents. Classes with fewer than two referents are eliminated. Then, for each class, the element that corresponds to the id that represents the class is chosen as a topic, unless the element is elaborated by another one. In that case, the elaborating expression is the one selected to be a topic. Figure 3.2 has a sentence example. The sentence is annotated, with <np> standing for noun phrase and <adp> standing for phrasal tag, indicating that its element cannot be the head of larger elements. Furthermore, <eq> is used to identify the id of the element that it refers to. In this case, "his" refers to the id "j0" that corresponds to the element "John", so the class would be named "John". In this way, the class would only have one referent which is not possible, but it serves here as an example.

However, there are topics more important than others, so in order to select the topics to utilize in the slides, spreading activation is applied to a tree with nodes as GDA elements

```
<np id="j0">John</np> beats
<adp eq="j0">his</adp> dog.
```

Figure 3.2: Sentence annotated with the GDA tagset. Image taken from Utiyama and Hasida [1999].

and links as their corresponding semantic relationships. Because the topics are GDA elements, they will each be assigned an activation value that reflects their importance. The topics with the highest values are the ones chosen for the slide decks. This process is explained in section 2.3.2.

The second stage is to create the presentation. The first slide will contain the selected topics, and each of the other slides corresponds to one topic: the title is the topic title, and the contents are the topic summary in bullet points. The content is composed of extracted sentences from the text that contain topical subjects. Redundant sentences are eliminated, and the remaining ones are itemized using defined heuristics. The slides are dynamically changed through the interaction of the audience. If during the presentation the audience wants to know something about a certain topic, and there is more information about it in the document, then a new slide is created regarding that topic.

The previous method is only possible due to the initial annotation of the content. Shibata and Kurohashi [2005] presented a system for the Japanese language that analyses raw text without the need for any previous annotation of the text. This system follows four steps: preprocessing the sentences, discourse analysis of the text, topic extraction, and the generation of slides.

In the first step, the sentences are separated into clauses, also referred to as discourse units. Afterwards, a discourse analysis of the text is conducted by detecting the relations between clauses in a sentence and between two sentences. Relations between clauses are determined through the calculation of the similarity between the clauses, while relations between sentences are obtained through the calculation of a score involving cue phrases, word/phrase chains, and similarity between two sentences.

With the document analyzed, the next step is to extract topics. Topics are words in a phrase whose head word is marked with "wa" due to characteristics of Japanese grammar, but in addition to that, a topic can also be extracted using cue phrases. If there are various cue phrases in a clause, the first one is going to be used to extract the topic. After having the topics, everything else is considered non-topics. Also, for certain predicates that have a specified case component, the non-topic parts are considered key points that will later have emphasis on the slides.

Finally, with all the analysis and identifications made, the only step left is to build the slides. These are built by following some heuristic rules: the title of the slide will be the title of the text if it exists, otherwise it will be the first topic found in the text; if there is a topic in the text, this will be shown at a first level and the following non-topics will be at a second level; otherwise only non-topic parts will be displayed; if these parts are key points, they will be emphasized; the level of a clause will be equal in the same sentence and the indent of a sentence will be determined according to the coherence relation to its parent.

A similar way of analysing the discourse structure of a document to generate slides was presented by Hanaue et al. [2012]. It represents the document as a network structure of units, considering a discourse unit as a text or image component, and the links between

them their respective semantic relationships. Then, using the discourse structure, a logical structure is created by grouping slide components according to their semantic relationship in order to obtain a layout template. The template will then convert the logical structure into a geometric structure, which is essentially the final layout of a slide, with figures, tables, text, and other elements in the proper places.

3.1.3 Graph

The last method [Hanaue et al., 2012] represented the data as a network with nodes and links connecting them. This section demonstrates a method in which a similar network is used, but, instead of using the discourse structure of the text, the summarization is accomplished through the construction of trees.

Sravanthi et al. [2009] presented a system that summarizes academic papers utilizing a query-specific summarizer – QueSTS [Sravanthi et al., 2008]. Firstly, an integrated graph (IG) is built. It is very similar to the network of the previous method, but instead of discourse units as nodes, it uses sentences, and rather than representing links as the semantic relationship, it uses some similarity scoring between sentences. Given a query, all node weights are calculated by adding the relevancy of the node to the query and the weights of neighbouring nodes.

After this, a contextual tree ($CTree$) is built for each query term from each node. Firstly, a node is selected and added to the tree, along with no more than three of its neighbors. Then, the tree continues to be built downwards, from the left to the right, until the maximum depth is reached or the last node contains a query term. In the end, the trees of all the query terms regarding a node are merged into a $SGraphr$. Figure 3.3 exemplifies the construction of the $CTrees$ from the node h of the integrated graph, taking into consideration two query terms, and their posterior merge into a $SGraphr$. The top side $CTree$ relates to the query one, that is in nodes b and f. Firstly, the node h and its neighbors g and d are added to the tree, and then, because b is closer to d than g, b is added as the child of d along with a and f. The bottom side $CTree$ concerns the second query, that is present in nodes g and d, and as such the construction of the tree stops in those nodes.

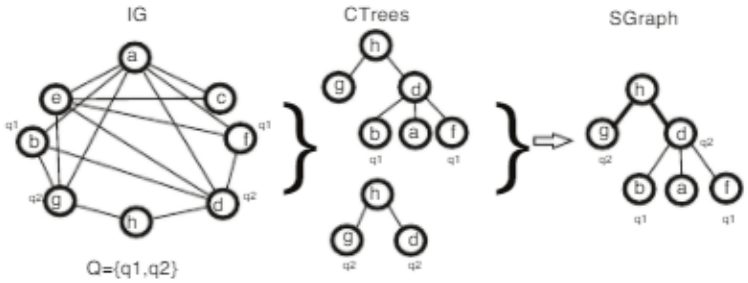


Figure 3.3: Example of the $CTrees$ and $SGraphr$ created from node h. Image taken from Sravanthi et al. [2009]

After having all the $SGraphs$, they are ranked through a scored model, with the edges belonging to all the $CTrees$ of a node counted more than once. The one with the highest rank is the summary.

However, the authors created a specific summarization method for every common sec-

tion of an academic paper, so this system is not used to summarize the entire document. All the methods are as follows:

- Introduction: The content of each of the sentence in this section is compared to the abstract, and the sentences with the highest similarity are put on the slides with the title equal to the title of the respective section.
- Related Works: The sentences with cite tags are retrieved and compared to the introduction. The ones with the highest similarity are placed on the slides.
- Model and Experiment: It gives the QueSTS keywords and sentences under the respective subsection. The sentences obtained are put on the slides in the order they appear on the paper.
- Conclusion: The keywords and the title of the paper are retrieved and then used as queries in the QueSTS system.

Furthermore, the paper is scanned for the presence of sentences referring to graphic elements ("for example..", "in the following equation.."...) and all these sentences are added along with the respective tag that contains said element.

3.1.4 Ontology

Ontologies are formal representations of a domain. Much like graphs, they consist of a set of concepts and relationships among them.

Mathivanan et al. [2009] presented a system, composed of 11 modules, that takes advantage of ontologies for scoring the sentences. The first module is a RTF Parser, in which bold, italics, and underlined text are parsed and converted to a text file that serves as input to the segmentation and ontology modules. In the first module, text segmentation is followed by segment chunking and noun phrase extraction. The second one creates an ontology and outputs a file in the Web Ontology Language (OWL) [Dean et al., 2004]. In the OWL parsing module, the OWL file is taken as input and parsed to create an ontology tree that is stored as an adjacency linked list. This is then used to find the semantic links and the physical (structural) links between sentences. The given output is a relational matrix.

Having all that information, sentence scoring is done next. The final sentence score is a weighted average of individual scores based on sentence position, sentence centrality, number of noun phrases in that sentence, number of keywords in the sentence and also in other sentences, and semantic link between the keywords in these sentences and keywords in other sentences.

As for sentence ranking, there are two different modules; one for title extraction and the other for bullet point generation. Candidates for the title are noun-phrases from important sentences or bold noun-phrases from the text, with the choice revolving around the noun phrase with the most semantic links to all other noun phrases. As for the content in the slides, sentences are put in Subject-Verb-Object format to generate the bullet points, which are further organized into Definition, Types, Examples, Advantages and Disadvantages by looking at each sentence's keywords.

Sathiyamurthy and Geetha [2012] utilized a very similar method, in which domain and pedagogy ontologies are constructed to extract semantic and contextual links from the

document for effective sentence and title extraction. The domain ontology aims to limit the use of a concept to a restricted sense, specific to the domain, and is composed of a domain layer, a class layer, which is a collection of concepts from the first layer, and a constraint layer that deals with relations and axioms of a concept in the second layer. The pedagogy ontology is an automatic annotation of contextual labels to the corresponding segments of text.

With those concepts, the system is ready to be built. This system has two main components, which are title identification and sentence selection. For the first component, given a text segment, a tree is constructed with the nodes representing domain concepts and the edges representing the hierarchy between the nodes. Then, in order to determine the title to be used within the domain concepts, the number of articulation nodes connecting to it is calculated. Nodes are articulation points if they have a child whose vertices are connected to the ancestors or, in the case of being a node root, have at least two children. The concept with the maximum number is the chosen title.

Finally, for sentence selection, a methodology very similar to Mathivanan et al. [2009]'s is used. The score, much like before, is based on centrality, layout position, number of noun phrases, number of keywords present in the sentence and also present in other sentences, and semantic link between the keywords in these sentences and those of other sentences identified using ontology.

3.1.5 Machine Learning

Supervised Machine Learning is currently the most used and documented method to the automatic generation of slides. Generally, this method is known for its use of several diverse classifiers. However, for this particular task, Support Vector Regression (SVR) is the most common choice in the literature. For instance, Hu and Wan [2013] used an SVR model, learned through the features and importance scores of the sentences in the training set, obtained through the maximum similarity between a sentence and the sentences in the slides, in order to assign the importance of each sentence in the test set. The importance of a sentence is the maximum cosine similarity between said sentence and each sentence on the corresponding slides, while the features of each sentence are:

- Similarity with the respective section and subsection titles.
- Number of words in the sentence that overlap with ones in the titles
- Sentence position: the position of a sentence in its section divided by the number of sentences in the section.
- Sentence's parsing tree information: the number of noun phrases and verb phrases, the number of sub-sentences, and the depth of the parsing tree.
- Stop words percentage.
- Length of a sentence and the number of non-stop words.

However, the authors only employ ML in the selection step. For the ranking of the selected sentences, Integer Linear Programming (ILP) is used instead. In this method, chunking is applied in order to obtain noun phrases that will be candidate key phrases. The key phrases are going to be the bullet points, and the sentences relevant to the phrases are placed below the bullet points. The objective function contains three parts that maximize:

the average importance of the sentences in the slides, the weights of the bigrams, which are sets of two words that appear on the slides, and the weight coverage of key phrases. Coverage refers to when a sentence contains the selected key phrases. Finally, the titles of slides are set by using the titles of the corresponding sections.

This work inspired many others, all using SVR for sentence scoring and ILP for sentence selection. Shaikh and Deshmukh [2016] used a simplified version, only making use of two features: word overlap and sentence position. Bhandare et al. [2016] used TF-IDF to calculate important sentences instead of cosine similarity. Syamili and Abraham [2017] included images, placing them in the slides respective to the section they are in the document.

However, SVR is not the only possible classifier for learning scores, nor is ILP the sole method for ranking. Sefid et al. [2019] presented a system that provides a Multi-Layer Perceptron (MLP) with three combined types of sentence embeddings in order to give an output score for a given sentence. Firstly, similar to the PPSGEN method, it starts by parsing the text with Stanford CoreNLP [Manning et al., 2014]. Then, in order to attribute a score to each sentence, instead of cosine similarity, the score is given by the maximum Jaccard similarity of unigrams and bigrams, sequences of one and two words, between a sentence and all the respective slides of the training data. After having the score, all the layers corresponding to the embeddings are ready to be built. The first layer corresponds to semantic embedding, meaning that the semantics of the sentence will be transformed into a vector. In order to obtain the said vector, two types of neural networks are tested, and the one with the best performance is chosen. The first one is a Convolutional Neural Network, in which the output is given by a merge of embeddings of all the bigrams in a sentence. Firstly, for each bigram, a *tanh* activation function is applied to create new features, receiving as inputs the bigram multiplied by a filter added to a bias. Then, max pooling is applied to all these functions in order to discover the best one that is going to be applied. Moreover, this system also applies a Recurrent Neural Network, composed by two networks that receive input word vectors previously trained. Both of these networks, LSTM and GRU, have the undertaking of selecting which information should be kept or forgotten.

The second layer stands for context embedding and simply creates two vectors with the semantic embedding of a sentence and its three previous and three following sentences in order to understand in which context a sentence stands.

Finally, the last layer makes use of the syntactic embedding. The layers above focus on semantic features, but this last one focuses on surface features such as: sentence section (belonging to sections that mention the general work has more weight), position in the section, number of noun phrases, number of verb phrases, number of sub-sentences, height of the parse tree, ratio of stop words among tokens, number of tokens, number of characters in the sentences, average IDF, average TF, and similarity of tokens of the sentence with tokens of the general and section titles. After having the scores, in order to select the sentences to be used in the summary, two common methods are used. Greedy search chooses the N best sentences that do not surpass a threshold of bigram overlap and summary size. ILP aims to maximize the product of a sentence score with the number of characters in the sentence.

Then, for the slide generation step in this system, each slide is composed of a title and two level bullet points. The second-level bullet point contains all the sentences previously extracted, and the first level contains phrases obtained through said sentences. Then, the phrases are ranked according to their average frequency and placed along with the sentences in the slides. Finally, the title of the slide is the title that corresponds to the

section of the first sentence in the said slide.

The same authors Sefid et al. [2021a] proposed an identical method, in which the sentence ranking and slide generation are the same, but the sentence labeling and scoring differ. Instead of labeling sentences with Jaccard, they use a windowed labeling ranking method, meaning that several windows will be chosen and, within them, sentences will be labeled according to the SummaRuNNer method [Nallapati et al., 2017]. This method assesses whether adding a given sentence will improve the ROUGE score, and if the answer is yes, then it is labeled with a one, otherwise it gets a zero.

As for the sentence scoring, instead of neural networks, this system uses novelty, importance, and position. However, in order to apply this, a previous step that creates a document embedding is required. This is done in two ways. One applies ReLU as an activation function that receives as input a multiplication of a weight, a bias, and the average of sentence encodings generated by a bi-LSTM and therefore composed of forward and backward hidden states of the last token in the sentence. The other one also uses sentence embedding, but instead of using only one token, all the hidden states of all the word-level tokens are concatenated to create the sentence encoding. Then, together with the model matrix to be learned, it is sent to a softmax layer to generate the attention weights. Finally, the sentence embedding is the average of the output of the softmax multiplied by the sentence encoding. This, along with a similar attention layer, is further used to create the document embeddings. Also, sentence level attention is used as a weight to identify the most important sentences.

After this, the only step left is to rank the sentences. So, it uses a sigmoid function that receives as input a sum of the position of the sentence in the document, the salience, the novelty, and the content similarity to the gold summaries. Salience makes use of document and sentence embeddings and tries to identify the importance of a given sentence. The novelty corresponds to how different a sentence is from the ones already in the summary. For that, it needs a summary embedding that is created by summing all the sentences already in the summary.

All the methods above focus on summarizing everything in a document at once. However, instead of doing this, Li et al. [2021] proposed a system that receives topics, described by section titles, as a query, and only summarizes the content related to each topic. For this purpose, the authors started by performing a study between academic papers and slide decks to determine the most important topics when creating presentations. This study consisted of determining the popularity of each topic and the portion of the text in the slides taken from the paper and not from an external source, and choosing the ones with the highest score. The results showed that the topics “Contribution", “Dataset", “Baseline", and “Future Work" were the most relevant and were, therefore, the ones picked.

With the topics chosen, the system proceeds to the slide generation, in which it starts by selecting the pseudo target and pseudo training data. The target is given by the output of a log-linear model with heuristic weights. For the data, the papers containing keywords related to the topics are chosen. After pre-processing, sentence scoring and selection are accomplished through mutual learning of the Neural Sentence Selection Model and Log-Linear Classifier with Prior Knowledge. Each classifier is trained alternately until they converge on each other.

The neural sentence selection model aims to capture the sentence semantics and is composed of two modules: Paper Encoder and Sentence Selector. The first module starts by creating a basic representation of each sentence in the paper using a single-layer bi-GRU. For each word in each sentence, it calculates the GRU hidden states and then concatenates

the last forward and first backward states to form the sentence representation. After that, the basic representation is applied to a bi-GRU and the results of the hidden states are concatenated to form the final sentence representation. Furthermore, after all the sentences are encoded, they are sent to the second module, which calculates the probability of each sentence being selected, through the use of GRU and functions softmax and *tanh*.

The Log-Linear Classifier with Prior Knowledge encodes the prior knowledge as features within a log-linear model and uses them to calculate the importance of a sentence. There are four features applied to each sentence with the following values:

- Keyword: the value is one for the sentences that have the keyword and zero for the ones that do not.
- Belonging Section: the value is one if the section of the sentence has a section keyword, otherwise is zero.
- Sentence Position: the value is the normalized position of a sentence.
- BERT-QA Signal: inputs a question related to the contents of each topic in the paper to the 2019 BERT-based Question Answering model [Devlin et al., 2019], fine-tuned on SQUAD [Rajpurkar et al., 2016]. Given the topic "Contributions" it return the question: "what are the contributions in this paper?". If the sentence contains the output, the value is one, otherwise it is zero.

All these papers presented methods that extracted entire sentences from a text. However, extractive retrieval is a major limitation in summarization due to the fact that, among other reasons, it is not possible to select the important part of a sentence. A sentence can have much useless information and be a crucial part of the summary. With that in mind, Wang et al. [2017] propose a methodology for extracting phrases instead of sentences. In their method, a Random Forest (RF) Classification model is trained to decide which phrases to include in the summary. Each sentence in the slides is parsed in order to find the verb and noun phrases (VP and NP), and then they are compared with the candidate phrases through cosine similarity. Furthermore, for each phrase, the following features are extracted: phrase position, phrase length, TF-IDF, section, phrase type (VP and NP), and parse tree information.

The output of the classifier will provide the salience of each phrase, and if it is greater than a threshold, it is kept. Then, each phrase should have the level of the bullet points of the slides assigned. To do so, the hierarchical relationship between phrases is determined, with only the most powerful relationships being kept. A RF classifier is used to identify the phrases that have that relationship. In this case, a method very similar to the one described above is used, but only the pairs of phrases in the slides that have a hierarchical relationship are used to compare the candidates. Moreover, there is the addition of the features related to the difference in position, phrase length, TF-IDF, type, and parse tree information. Only the phrases with the strongest hierarchical relations are kept. Finally, in order to select the phrases, two different greedy algorithms are proposed. One selects phrase pairs with strong hierarchical relationships, and the other selects individual phrases with high salience.

Even though this system is a step forward to better summarization, there are still many limitations, like, for example, adding new words, since it is not possible to derail from the given phrases. For that, abstractive summarization is exploited.

3.2 Abstractive summarization

Fu et al. [2021] proposed a system that, at first glance, seems identical to the extractive methods. In fact, extractive methods are used to select important sentences. However, after selecting those sentences, an abstractive method is applied to paraphrase them and making them more concise. Text box 3.2.1 shows an example of a paraphrased sentence.

Original:
The Carnation Revolution, also known as the 25 April, was a military coup by left-leaning military officers that overthrew the authoritarian Estado Novo regime on 25 April 1974 in Lisbon.

Paraphrased:
The authoritarian Estado Novo regime was overthrown in Lisbon on April 25, 1974, during the Carnation Revolution, also known as the 25 April, which was a military coup led by left-leaning military officers.

Text 3.2.1: Original and paraphrased sentence extracted and edited from the Wikipedia article "Carnation Revolution".

So, the system is composed of four modules: Document Reader (DR), Progress Tracker (PT), Object Placer (OP) and Paraphraser (PAR). DR encodes the figures and text. The text is encoded sentence by sentence with the transformer RoBERTa [Liu et al., 2019] and then, a contextualized sentence embedding is extracted using a Bidirectional Gated Recurrent Unit. As for the figures, embeddings for the image and caption are created with the network ResNet-152 [He et al., 2016] and then combined into a figure embedding. After that, both text and figure embeddings are projected to a shared embedding space using a two-layer MLP and combined with a section embedding. Then, the output is sent to PT that has pointers for the slides and the sections and learns a policy to progress to the next section or slides. It is composed of a three-layer hierarchical RNN, in which each layer encodes the latent space for each level in a section-slide-object hierarchy. The first layer uses GRU to encode the information of a section. The second layer uses GRU and a two-layer MLP that learns a policy to predict if more slides for that section are required or if they should advance to the next section, and the third layer uses the same method as the second layer, but this time the policy determines if the object should be in the current slide or not, which is, to some extent, similar to the next section OP that selects objects from sections and decides on which slide and in which position they should be, also using MLP. Finally, the PAR module is where the abstractive method of this system is represented. It takes a sentence, before placing it in the slides, and paraphrases it using an attention-based sequence-to-sequence. So, in reality, the abstractive part is quite small, as most of the system is composed of extractive methods.

Sun et al. [2021a] presented a methodology that only uses abstractive methods, by taking automatic generation of slides as a question-answering problem. The idea is to have a title for each slide, and then the document is queried for content related to the title or its keywords. For that, it resorts to four modules. In the Keyword Module, a hierarchical tree of the titles of the document is constructed. Then, in the Dense IR Module, a dense vector IR model is trained to minimize the cross-entropy loss of titles to their original content. This model will be later used to compute representations for all paper snippets and slide titles. After that, in order to measure the previous vectors' similarity, a pairwise inner product between them is applied. The snippets with the highest similarity are chosen as inputs for the next module. Finally, in the QA module, there is a BART model [Lewis

et al., 2019] that uses the slide title and related keywords as a query and the top-ranked text snippets as context. Furthermore, there may be a figures and tables module that employs the dense vector IR model to determine the degree of similarity between captions and slide titles, with the most similar ones being used.

3.3 Other summarization Methods

So far, this chapter has only presented summarization methods that were used in the state of the art with the objective of creating slides. However, there are other methods that were not included but could also be interesting to apply to a slide generation problem. This section presents some, among them TextRank [Mihalcea and Tarau, 2004], LexRank [Erkan and Radev, 2004], LSA [Deerwester et al., 1990] and Lexical Chains [Sethi et al., 2017].

TextRank [Mihalcea and Tarau, 2004] is a method that resorts to graphs, such as QueSTS, in order to automatically summarise one or several documents. This method starts by, in the case of existing documents, grouping them into the same text. Then, it splits the text into sentences and transforms them into vectors. Through those vectors, it calculates the similarity between sentences and puts the results in a matrix that will be converted into a graph, where nodes are sentences, and edge weights are the similarity of the sentences it connects. Finally, based on edge weights, a ranking algorithm is applied to the graphs, and the sentences with the highest ranking are the ones chosen for the summary.

LexRank [Erkan and Radev, 2004] is similar to TextRank, with the difference being in the selection of the most important sentences. For that, this method utilises the eigenvector of centrality. At the end of this operation, the sentences with the highest values are the most important and the ones chosen.

LSA [Deerwester et al., 1990] analyses relationships between a set of documents and the terms contained within. For that, it creates a term-document matrix, where each cell corresponds to the frequency of a word or term in a document. Then the matrix is given as input to the Singular Value Decomposition (SVD), that will output three matrices, that will be used by some method in order to select the sentences for the summary. In these experiments, the Steinberger and Jezek [2004] method was the one used. Steinberger and Jezek [2004] makes use of the matrix concepts x sentences, where the rows are sorted by the importance of the concepts. With this matrix, the length of the sentences is determined by considering concepts with indices less than or equal to the provided dimension. Then, as a multiplication parameter, a second matrix resulting from STD is employed to emphasise the most significant concepts. The chosen sentences for the summary are the ones with the longest lengths.

Lexical Chains [Sethi et al., 2017] are sequences of semantic related words. In order to understand how words were connected in this method, the lexical database WordNet [Fellbaum, 1998] was used, more specifically, the synsets of each word were obtained, were the synsets correspond to the various meanings associated with the respective word, and then the lexical chains were constructed, where each chain corresponded to a meaning and the words that composed them were words with that meaning. However, not every word was used, only nouns and proper nouns were considered. It is also important to mention that, through the use of coreference resolution, for every pronoun the respective noun was obtained in order to have a better grasp of the frequency/importance of each of them.

After having the lexical chains, the next step was to assign a score to each sentence. For each chain that respects the following equation (strong chain), the first sentence containing this chain is added to the summary:

$$Score(Chain) > AVG(Scores) + ratio \times STD(Scores) \tag{3.1}$$

were the score of a chain is given by:

$$Score(Chain) = Length * (1 - \frac{Isolated}{Length}) \tag{3.2}$$

with length being the sum of the frequencies of every word of the chain in the text, and the variable $Isolated$ is equal to the number of words in the chain that only occur once in the chain.

All the sentences with a score (see equation 3.3) greater than the average of sentences scores are also added to the summary:

$$Score(Sentence) = \frac{Count(Nounsinstrongchains)}{SentenceLength} \tag{3.3}$$

The sentences were added to the summary in the order they appear on the original article.

In addition to these methods, this problem may also benefit from the use of abstractive summarization methods. In order to test these other methods, transformers were used. Transformers are machine learning models that are already implemented and ready to use. These are pre-trained, i.e., trained on a large generic corpus and then fine-tuned, i.e., adapted to a particular task or dataset, in this case summarization. However, these models have a limitation, since they can only process small texts, resulting into bigger texts being cut for the models to function.

Seven different summarization transformers were tested for familiarisation purposes, to analyse their results and determine their relevance to the project. These transformers can be found in the HuggingFace library [1]. They are:

- *T5 Large For Text Aggregation* [Pletenev, 2021]

  Raffel et al. [2020] proposed a transformer (T5) that is capable of solving any NLP task with the same model, loss function, and hyperparameters. This is possible because T5 reframes all tasks to a text-to-text-format where the input and output are always text strings.

  Furthermore, the authors also proposed a dataset (C4), used for pre-training, composed by a clean version of Common Crawl[2]. This cleaning involved deduplication, discarding incomplete sentences, and removing offensive or noisy content.

  Pletenev [2021] employs T5 and fine-tunes the model with the CrowdSpeech dataset [Pavlichenko et al., 2021]. The texts in this dataset are broken down into audio files using speech synthesis tools, which are then distributed to crowd workers to annotate, resulting in the creation of the gold standard summaries.

[1] https://huggingface.co
[2] https://commoncrawl.org

- *Multilingual T5* [Hasan et al., 2021]

  mT5 is a multilingual variant of T5 that was pre-trained on mC4, a variant of C4 that has text in 101 languages. However, it does not include any supervised training. Therefore, it needs to be fine-tuned to be useable on a downstream task. In this transformer, the XLSum dataset was used for this purpose.

  [Hasan et al., 2021] assembled a dataset formed by over 1 million articles written in 44 languages, extracted from the BBC website. This dataset is highly abstractive and concise. Its extraction was made through a curation tool that automatically extracts the articles and summaries from BBC. These summaries are composed of one or two sentences, written by the authors to give an outline of the whole article.

- *Roberta2Roberta L-24* [Rothe et al., 2019]

  This transformer was initialized on the Roberta-large checkpoints and was fine-tuned using the XSum [Narayan et al., 2018] dataset. This dataset is composed of BBC articles and their brief summaries written by the authors.

  RoBERTa follows the implementation of BERT but modifies some hyperparameters, training with much larger mini-batches and learning rates. This model was pre-trained with the following datasets: BOOKCORPUS [Zhu et al., 2015] (used in BERT), CC-NEWS [Nagel, 2016] (news taken by Common Crawl), OPENWEBTEXT [Gokaslan and Cohen., 2019] (text is web content extracted from URLs shared on Reddit) and STORIES Zhu et al. [2015] (extracted through Common Crawl).

- *Distilbart 12-6* [Shleifer and Rush, 2020]

  This transformer is a distillation version of BART, i.e., a shrunken version of the model, that is re-fine-tuned with, in this case, the CNN/DailyMail dataset.

  BART [Lewis et al., 2019] is a model that was pre-trained using the following datasets: SQuAD 1.1 [Rajpurkar et al., 2016], MNLI [Williams et al., 2017], ELI5 [Fan et al., 2019], XSum [Narayan et al., 2018], ConvAI2 [Dinan et al., 2019], and CNN/DailyMail [Hermann et al., 2015, See et al., 2017]. Due to the diversity of the datasets, this model is capable of several tasks, such as, summarization, translation, dialogue...

- *Pegasus Wikihow*, *Pegasus CNN/Dailymail*, *Pegasus Xsum*

  These last three transformers [Zhang et al., 2019] belong to the same model, which was trained on the C4 dataset [Raffel et al., 2019], and the HugeNews dataset, a collection of 1.5 billion articles that the authors compiled themselves published between 2013 and 2019, including articles from the XSum [Narayan et al., 2018] and CNN/DailyMail [Hermann et al., 2015, See et al., 2017] datasets. The difference between these transformers lies in the dataset used for fine-tuning each model. Finally, each of them was tested on the testing set of the C4 and HugeNews datasets.

  The WikiHow dataset [Koupaee and Wang, 2018] is constructed by concatenating all the paragraphs to form the text and by concatenating all the bold lines at the beginning of each paragraph to form the golden standard summaries.

These transformers were employed using the HuggingFace library [3]. This library is composed by several models and datasets in several NLP problems, being one of them summarization.

[3] https://huggingface.co

3.4 Summary

In this chapter, several state-of-the-art research projects for slide generation were identified and briefly explained. The algorithms used for the creation of slides were all composed of two main steps: summarization and slide generation, with the first step receiving the most attention because it focuses more on the slides' text and allows for the application of a variety of different methods. These methods can be extractive or abstractive, with very little research supporting the latter. Extractive methods extract full sentences from the original text and place them in the summary, while abstractive methods try to create summaries more like what a human would do, adding new sentences or words to the summary.

There are five types of research for extractive methods: statistical, discourse-based, graph, ontology, and machine learning. From those types, only machine learning is supervised, meaning that these approaches need to be trained, contrarily to the others that are unsupervised, and so they do not require any dataset for training, allowing for a summarization of every text, independent of their language or topic. As for the abstractive methods, there are only two papers that research it. One simply uses extractive methods and then applies paraphrasing to the resulting summary, while the other uses a question-answering model that takes section titles as questions and creates answers for them.

The table below (3.1) shows a brief synthesis of the works described in the state of the art for slide generation. For each work, the used evaluation metrics, resources, and methods are identified, also presenting brief highlights of the paper's content. This will make it easier to pinpoint the several differences between methods.

Citation	**method**	**Resources**	**Key Aspects**	**Evaluation**
Sariki et al. [2014]	Extractive: Statistical	N.D.	Merger of the statistical methods: cue-phrase, word frequency, title similarity, location	Similarity with manual and automatic summaries
Utiyama and Hasida [1999]	Extractive: Discourse Based	GDA tagset [Nagao and Hasida, 1998]	Annotation of the syntatic structure, semantic relations and coreference, identification of topics, dynamic adaptation of the slides	Human
Shibata and Kurohashi [2005]	Extractive: Discourse Based	JUMAN [KUROHASHI, 1994], KNP [Kurohashi and Nagao, 1994]	Discourse analysis of the text: detection of relation between clauses in a sentence and between two sentences, topic extraction and generation of slides through heuristic rules	Human
Hanaue et al. [2012]	Extractive: Discourse Based	N.D.	Construction of a network structure with nodes as text and images and links as their semantic relationship. Then accordingly to that relationship group slides components	N.D.
Sravanthi et al. [2009]	Extractive: Graph	LaTeXML, QueSTS [Sravanthi et al., 2008]	Composed by an integrated graph in which the nodes are sentences and the edges represent the cosine similarity between the nodes. Key phrases are extracted and given as a query to the graph in order to extract important sentences related to it	Human
Mathivanan et al. [2009]	Extractive - Ontology	Doddle OWL [Morita et al., 2006]	Text segmentation, chunking, extraction of noun-phrases and ontology creation to identify important sentences	Precision, Recall, F1-Score

Sathiyamurthy and Geetha [2012]	Extractive - Ontology	ACM Computing Classification System [4]	Domain and pedagogy ontology creation to identify important sentences	Precision, Recall, F1-Score
Hu and Wan [2013]	Extractive: Machine Learning	PDFlib [5], ParsCit [Councill et al., 2008], xpdf [6], Microsoft Office API [7], OpenNLP [8], Arnetminer [9] IBM CPLEX optimizer [10]	Prediction of importance scores for each sentence using SVR, selection of the most important contents using ILP	ROUGE-1, ROUGE-2, ROUGE-SU4 and T-Test
Bhandare et al. [2016]	Extractive: Machine Learning	N.D.	Applies SVR model for important sentence calculation, applies ILP model for generalization of slide	ROUGE
Shaikh and Deshmukh [2016]	Extractive: Machine Learning	Stanford NLP [Manning et al., 2014]	Measures sentence importance with SVR. Selects the best ones with ILP	ROUGE-1 ROUGE-2 ROUGE-SU
Syamili and Abraham [2017]	Extractive: Machine Learning	N.D.	Uses SVR for sentence scoring and ILP for selection	N.D
Wang et al. [2017]	Extractive: Machine Learning	N.D.	Uses a random forest classification model to select the candidate phrase and determine the hierarchical relation between phrases. The phrases are selected through a relation-first and a saliency-first algorithm	ROUGE-N
Sefid et al. [2019]	Extractive: Machine Learning	Stanford CoreNLP [Manning et al., 2014]	Label the sentences, rank the sentences using semantic, context and syntactic embedding, select the sentences through a greedy and ILP method and generate the slides	ROUGE-1 ROUGE-2 ROUGE-L ROUGE-W
Sefid et al. [2021a]	Extractive: Machine Learning	PS5K dataset, SummaRuNNer [Nallapati et al., 2017], Stanford CoreNLP [Manning et al., 2014]	Label the sentences, rank the sentences using novelty, importance and position, select the sentences through a greedy and ILP method and generate the slides	ROUGE-1 ROUGE-2 ROUGE-L
Li et al. [2021]	Extractive: Machine Learning	ACL Anthology Reference Corpus [Bird et al., 2008]	Selection of sentences based on a topic. Sentence scoring and selection is accomplished through mutual learning of Neural Sentence Selection Model and Log-Linear Classifier with Prior Knowledge	Performance of relevant sentence selection from paper and comparison with human-generated slide

[4] https://www.acm.org/publications/computing-classification-system/1998/ccs98
[5] https://www.pdflib.com
[6] http://www.xpdfreader.com
[7] https://docs.microsoft.com/en-us/office/dev/add-ins/powerpoint/
[8] https://opennlp.apache.org
[9] https://www.aminer.org
[10] https://www.ibm.com/products?types[0]=software

Fu et al. [2021]	Extractive: Machine Learning and Abstractive	SciDuet dataset, ResNet-152 [He et al., 2016]	Composed by 4 modules: Document Reader encodes figures and text, Progress tracker learns a policy to progress to the next section or slides, Object Placer selects objects of sections and chooses in which slide they should be and in which position and Paraphraser paraphrases sentences	ROUGE-SL, LC-FS, mIoU, TFR, Human
Sun et al. [2021a]	Abstractive: Query-based	DOC2PPT dataset	Users input the slide title and then the document is queried for content related to the title or its keywords	ROUGE metrics and qualitative Human

Table 3.1: Compendium of the automatic slide generation papers

Furthermore, there is a last section in this chapter that describes other summarization methods that were not used for the slide generation problem but that might be interesting. Among them are four extractive methods (TextRank, LexRank, LSA and Lexical Chains) and seven transformers (models already trained) for abstractive summarization. Even though transformers are used for several NLP problems, it is still not possible to apply them to a slide generation problem because slide decks have several aspects that need to be evaluated, such as information organization, figure inclusion and placement, among others, that transformers cannot yet process. Therefore, the next best option is to only evaluate the text, i.e. the summary.

Chapter 4

Summarization Results

The main component of this work is the summarization of texts. Therefore, as a first step experiments regarding only this step were conducted. The outcomes of all those experiments are presented in this chapter. These tests involved the use of automated evaluation metrics, more specifically, ROUGE, BERTScore, and BLEURT, to evaluate extractive and abstractive summarization methods when applied to two datasets composed of text and slides: SciDuet [Sun et al., 2021b] and PS5K [Sefid et al., 2021b].

The first section of this chapter identifies all the methods used for this report's experiments, while the second describes the used datasets. The next two sections present, respectively, all the results obtained for the extractive and abstractive methods. In each section, a discussion of the results is given, and then in the fifth section, a comparison is made between both types of summarization. Furthermore, there is a last section that summarises all the main conclusions drawn from the experiments.

4.1 Methods

As already mentioned, methods for summarization can be of two types: extractive or abstractive. To have a good understanding of which one is more appropriate for the final objective of constructing slides, both options were explored. For extractive, seven methods were chosen: LSA [Deerwester et al., 1990], Lexical Chains [Sethi et al., 2017], TextRank [Mihalcea and Tarau, 2004], LexRank [Erkan and Radev, 2004], TF-IDF [Jones, 1972, Luhn, 1958], QueSTS [Sravanthi et al., 2009], and SVR [Hu and Wan, 2013]. The last two were chosen because they were already applied in previous research and had promising results, as is stated in sections 3.1.3 and 3.1.5 of chapter 3. The implementation of QueSTS followed the details presented in section 3.1.3. The only difference lies in the fact that this method was used to summarise all the text of a document, while Sravanthi et al. [2009] used different methods to summarise the different sections of the academic papers (no other type of documents were considered). As for the SVR method, the only supervised extractive method, its implementation followed the work described in Hu and Wan [2013], with some changes to the sentence selection step. Integer Linear Programming is also used, but the objective function only seeks to maximise the average importance of the sentences in the summary, having as a limitation the number of sentences that the summary can have.

Regarding the other methods, they were chosen due to their frequent use in summarization problems but uncommon use in summarization for slide generation. TF-IDF [Jones,

1972, Luhn, 1958] is a straightforward, frequency-driven method, as described in section 2.3.1. For summarization, this method is applied to every non-stop word in every sentence, but instead of TF calculating the frequency of a word in a document, it calculates the frequency of a word in a sentence, and instead of IDF receiving the number of documents, it receives the number of sentences in the document. Then, the value of TF-IDF of all words in a sentence is added, and the sentences with the highest score are the ones chosen.

The remaining methods, TextRank, LexRank, LSA, and Lexical Chains, are applied as described in section 3.3.

All of these methods are extractive, which means that the summary they produce is composed of sentences extracted exactly as they are from the original text. As an attempt to improve the quality of the summaries, a new method was created that combines the summaries obtained from the different methods. This is done using sentence frequency, i.e., counting the number of occurrences of each sentence in the summaries of the different methods and constructing/building the final summary from the sentences with counts above a certain threshold. This aims to find the best sentences, supposing that the more summaries a sentence appears in, the more important it is.

Additionally, abstractive methods using transformers were investigated in addition to these extractive methods, as detailed in section 3.3.

4.2 Datasets

The approaches studied in this work all seek to summarise texts. However, the final objective is not to create summaries but to create presentation slides. With that in mind, summarization methods were tested in datasets used for slide generation. So, even though it is not possible to access the quality of the slides, it is possible to access the quality of the text that will be later included in them. If these approaches were to be tested with datasets of summarization, it would still be possible to understand the quality of the summaries. However, summaries made for slides versus those made for summarization only can differ. For instance, it is common to present information in a slide as a set of short phrases (or bullet points), whereas the relevant, or more detailed, information related to the phrases is filled in by the speaker during the presentation. This is unlike a text summary which condenses the original text into a shorter amount of text while maintaining all of the important information.

With that in mind, two datasets were tested: SciDuet [Sun et al., 2021b] and PS5K [Sefid et al., 2021b] which include published scientific papers paired with human-produced slides used for presenting them. The datasets contain papers in the fields of computer and information science, machine learning, neural information processing systems, and computational linguistics.

SciDuet has 952–55–81 paper-slide pairs in the Train–Dev–Test split, while PS5K has 4000-250-250. To use these datasets, the text from the documents is used as the input to the methods. The text of the slides is also extracted and used as a reference, against which the produced summaries are evaluated. Figure 4.1 show some of the slides used as reference and the text that is extracted from them.

In addition to these datasets, the CNN/Daily Mail dataset [Hermann et al., 2015, See et al., 2017] was also used. This dataset is summarization only, meaning that it is composed of documents and their respective summaries instead of presentation slides (that will have their text extracted and joined as a summary). As a result, the goal summaries for this

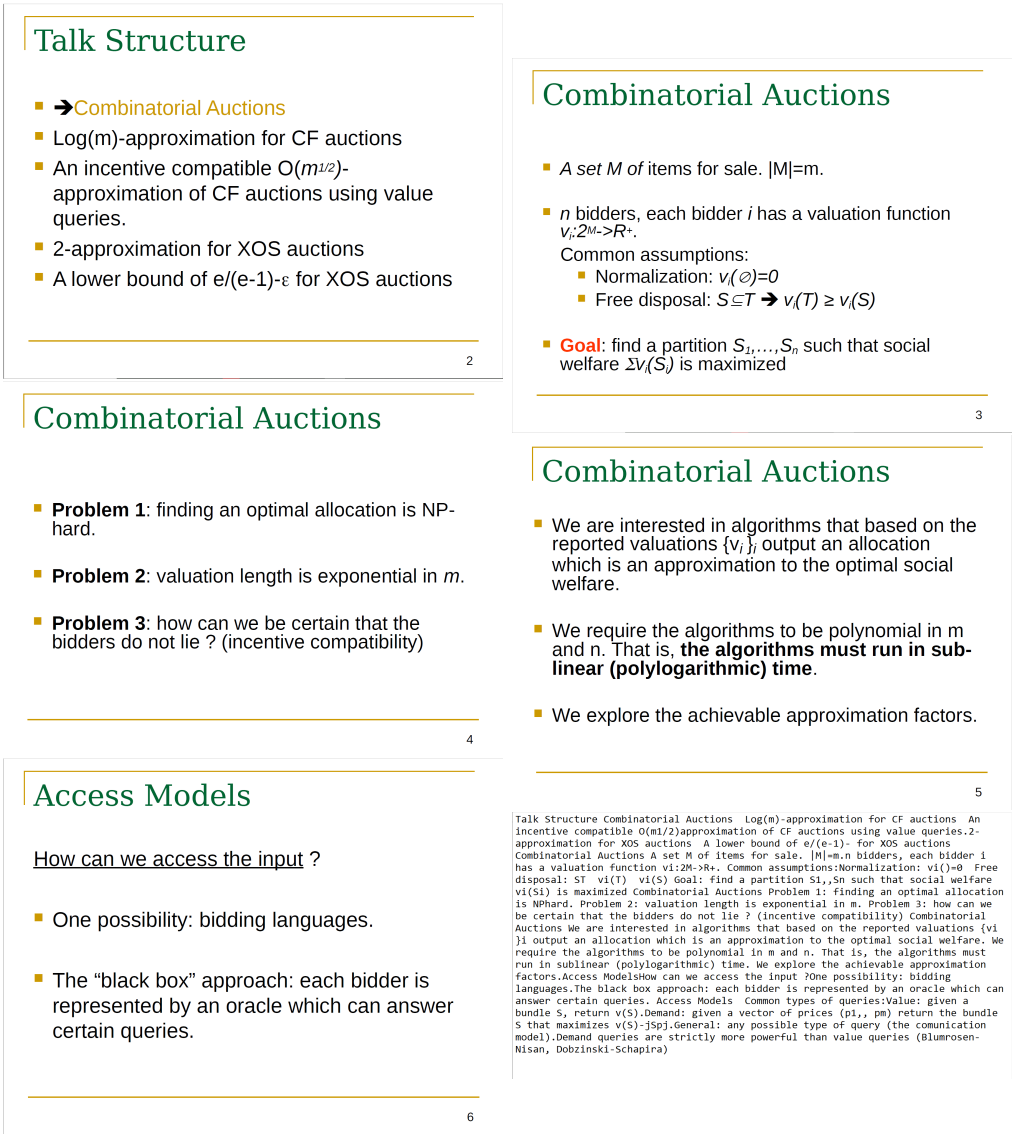


Figure 4.1: Set of Slides taken from the dataset PS5K relative to paper "Approximation Algorithms for Combinatorial Auctions with Complement-Free Bidders" from authors Shahar Dobzinski, Noam Nisan and Michael Schapira, and the sentences that are extracted from each slide.

dataset are smaller, which is very helpful for transformers since they can only process short texts. The PS5K and SciDuet, on the other hand, would require text to be cut in order for the transformers to use them, which would not be ideal because a lot of crucial information would be lost. Therefore, the CNN/Daily Mail dataset was used to test transformers. This is an English-language dataset with articles written by journalists at CNN and the Daily Mail and their respective IDs and highlights. The data has three splits: train, validation, and test, each one composed of 287.113, 13.368, and 11.490 instances, respectively. Due to the size of the dataset, only the first 400 articles and corresponding highlights from the testing split were used. Text box 4.2.1 shows an example of this dataset.

Article:
(CNN)Pakistan's highest court Friday ordered the release of Zaki-ur-Rehman Lakhvi, the alleged mastermind behind the Mumbai attacks, calling his detention illegal. Lakhvi, a top leader of the terrorist group Lashkar-e-Taiba, was not present at Friday's court proceeding. The terror attacks in India left more than 160 people dead in November 2008. In the attacks, heavily armed men stormed landmark buildings around Mumbai, including luxury hotels, the city's historic Victoria Terminus train station and a Jewish cultural center. On Friday, India summoned the Pakistan high commissioner "to convey our strong feelings about (the) Lakhvi verdict," said India's external affairs spokesman Syed Akbaruddin. Last year, the court granted Lakhvi bail, a decision the Pakistani government had said it would challenge. Many in India are still angry over the attacks and had criticized the bail decision. "It is very disappointing that the accused of the Mumbai attacks has been granted bail," the nation's home minister, Rajnath Singh, said in December. India executed the last surviving gunman from the attacks in 2012. Other suspects were all killed during the series of attacks, which went on for three days. CNN's Harmeet Shah Singh contributed to this report.

Summary:
The terror attacks in India left more than 160 people dead . A court granted the suspect bail last year .

Text 4.2.1: Example of CNN/DailMail dataset

All the methods in this report where tested using the Test split of the datasets. The methods that needed training were trained using the Train split.

4.3 Implementation

The implementation of the methods and the extraction of text from the datasets and from Wikipedia was done using the programming language Python. The SVR, TF-IDF, Lexical Chains, and QueSTS were the only methods mentioned in section 4.1 that were implemented entirely from scratch; the others had pre-implementations from various libraries. LSA and TextRank were tested using sumy library[1]; LexRank used LexRank library[2]; and the transformers were applied through HuggingFace[3].

Furthermore, other libraries were used in order to preprocess the texts, help in the implementation of the methods, and in the construction of the Wikipedia dataset and the

[1]https://github.com/miso-belica/sumy
[2]https://github.com/crabcamp/lexrank
[3]https://huggingface.co

slide decks. Among them are nltk[4], numpy[5], wikipediaapi[6] and pptx[7].

As for the evaluation metrics these were applied using the libraries: rouge-metric[8], bert-score[9] and bleurt[10].

4.4 Extractive Methods

This section presents the seven extractive methods identified. Six of those methods are unsupervised: Latent Semantic Analysis (LSA) [Deerwester et al., 1990], Lexical Chains (LC) [Sethi et al., 2017], TextRank [Mihalcea and Tarau, 2004], LexRank [Erkan and Radev, 2004], QueSTS [Sravanthi et al., 2009], and TF-IDF [Jones, 1972, Luhn, 1958] and thus require no training data. Support Vector Regression (SVR) [Hu and Wan, 2013] is supervised.

For each dataset and method, tests were conducted with different ratios, the lower the ratio, the shorter the summary. Every metric was computed for the same ratio values, except for TF-IDF, where sentences with a higher score than the average are put in the summary, regardless of their size; Lexical Chains (LC), where the ratio is a multiplication value used to determine the value that a chain must have to be considered strong; and for QueSTS, where the ratio corresponds to the number of leaves a tree can have. For QueSTS, the lowest ratio (LR) is 3 leafs and the highest ratio (HR) is 11. For Lexical Chains, LR is 0.5 and HR is 1. For the other methods, LR is 0.1 and HR is 0.4.

For the best-performing ratios, the metrics were also computed after preprocessing both the outputs and the gold summaries. This involved the following: removal of stop words and punctuation signs; stemming of words; and case folding. This was done to keep only the relevant words in the sentence and to minimise the problem of two words not being considered similar, even though at their root they are. Results for preprocessing are identified by 'Pre'.

Furthermore, as a way of trying to improve the results, several methods were combined. Due to space restrictions, in the tables of the two following subsections, the text of the column "Method" follows the pattern A-B:C, where A is a threshold that represents the number of summaries a sentence needs to be in to be chosen for the final summary, and B:C are numbers that represent the methods combined. A number is given for each method. TF-IDF-1, TextRank-2, QueSTS-3, LexRank-4, LSA-5, Lexical Chains-6, and SVR-7. The ":" denotes "to", so in A-B:C the methods B through C with a threshold of A are involved. "," which means "and" could be used in place of ":".

4.4.1 Results in PS5K

The tables 4.1, 4.2 and 4.3 display the ROUGE scores, while table 4.4 displays the BERTScore and BLEURT scores of all extractive methods in the dataset PS5K in a low and high ratio and with and without preprocessing.

[4]https://www.nltk.org
[5]https://numpy.org
[6]https://pypi.org/project/Wikipedia-API/
[7]https://github.com/scanny/python-pptx
[8]https://github.com/li-plus/rouge-metric
[9]https://github.com/Tiiiger/bert$_score$
[10]https://github.com/google-research/bleurt

The disparity between the lowest and highest ratios is evident when initially examining the ROUGE scores, with the lowest ratios typically displaying superior precision while having the worst recall. Due to the precision of methods with higher ratios typically being very subpar, the F-score is better for methods with lower ratios.

For example, looking at the scores of LexRank (not preprocessed), we see that LR has greater precision than HR, which allowed for a higher F-score. However, recall is lower. This shows that even though a higher ratio results in a greater fraction of relevant sentences being retrieved, many are not relevant. So, despite having fewer relevant sentences, the shortest summaries end up having higher relevance as a whole. This is true for every metric, and for every method but LSA in the ROUGE-S metric, which has a slightly better F-Score, LC, which has the worst precision and F-Score with a lower ratio, and QueSTS.

Contrary to other approaches, in QueSTS, a higher ratio leads to a better summary. This happens because this method handles the ratio differently: a low ratio here corresponds to a shorter summary than for the other methods. So, while the summary in the other methods has a good size, in QueSTS it is too short, and so needs a higher ratio, even though the summaries with the higher ratio can still be pretty small. This is because the ratio is not a percentage of the original summary that is kept but the maximum number of leaves a graph is allowed to have, which can still generate small summaries. Furthermore, QueSTS presents, mostly, the worst recall and the best precision of all the methods researched. The low recall means that the percentage of relevant sentences retrieved was low, which can mean that the summary is too small, which demonstrates what was said before. On the other hand, the precision is the highest of all the methods, which means that QueSTS is better at finding the most important sentences. This can also be a consequence of the summaries being small, i.e., this method, since it does not have a quota of sentences that need to belong in the final summary, only selects the most important sentences, disregarding other sentences that are less important, and therefore creates a small but highly relevant summary. However, for the final objective of constructing slides, a more informative summary might be more pertinent.

Similar to QueSTS, Lexical Chains exhibit different behaviour due to their different ratios. Contrarily to other methods, in ROUGE, for a lower ratio, this method presents a better recall than precision, resulting in a worse F-Score than the one of a higher ratio. So, in order to have a summary that is more similar to the target, the ratio must be higher, though this does not imply that it must be higher than the other methods because the ratio is different.

Regarding the BERTScore, all methods and ratios consistently display a higher recall than precision, with every method except LSA having a better F-Score with a higher ratio. BLEURT on the other hand, performs better with a higher ratio for every method, except TextRank and Lexical Chains.

In addition to testing the methods with a high and low ratio, an experiment was also performed where preprocessing was applied for every method in their best performing ratio. This was able to improve ROUGE-1, some values of ROUGE-SU, and BERTScore, except for the Lexical Chains method. ROUGE-1 compares words separately and so having stemming will relax the exact word match requirement to allow matching of different word forms, i.e, words that were identified as different may have the same root and with stemming they will be considered the same which will improve the score. ROUGE-SU is a mix of ROUGE-1 and ROUGE-S, so the impact of preprocessing on ROUGE-1 is also felt in this metric. As for BERTScore, stemming might have improved the contextual embeddings therefore improved the final score.

However, the preprocessing has a negative impact on the other metrics. This is mainly due to the removal of stop words. For example, ROUGE-L measures the longest common subsequence of words and removing words that are used often will shorten those sequences and consequently impact the score (negatively). Having stop words could add one or two words to a sequence which would improve its score. Something similar happens to the other metrics.

Analyzing the results presented in tables 4.1 , 4.2, 4.3 and 4.4 shows that the best method for ROUGE-1 is SVR Pre (or SVR LR and LSA LR without preprocessing), followed closely by TextRank LR, for ROUGE-2 the best is TextRank LR and LC HR, for ROUGE-3 the best is LC HR, and for ROUGE-4 the best is LC HR and TextRank LR. For ROUGE-L, W, S and S the best method is TextRank LR. As for BERTScore the best is TextRank Pre and LSA Pre (or TextRank HR without preprocessing). Finally, for BLEURT the best performing method is LexRank HR (followed closely by LC LR and TextRank LR). The results show that for this dataset the best performing method was TextRank. This method was the best in 70% of the cases and was very close to the best in the remaining cases. The difference between TextRank and the best performing in ROUGE-1 is only 0.09 with preprocessing and 0.05 without, in ROUGE-3 is 0.03 and in BLEURT is 0.02.

This conclusion is a little unexpected. It was anticipated that SVR would perform better because it is a supervised method, which means it was trained with the dataset and thus already has some knowledge of it. However, this is not what happens. While SVR is the best method for ROUGE-1, for the other metrics its ranking can be very diverse, oscillating from second to fourth best, with the exception of BLEURT, where it is the worst performing method.

Furthermore, the methods of the related work [Sefid et al., 2021b] are also all supervised. The ROUGE-1 and 2 scores achieved are all below the best supervised method reported for PS5K in related work, respectively 0.48 and 0.12. For ROUGE-L, however, the best reported score is 0.238, which is in line with LSA LR (0.235) and LexRank LR (0.233) and is outperformed by TextRank LR (0.247). So, TextRank is able to outperform the related work in 1/3 metrics and SVR in all metrics (except for when the text is preprocessed), which suggests that unsupervised methods should be regarded as interesting alternatives to explore in slide generation.

A variety of methods were combined in an effort to enhance the outcomes. Tables 4.5, 4.6 and 4.7 present all ROUGE values obtained for the several combinations, while table 4.8 presents the BERTScore and BLEURT.

In these experiments, several combinations were made, but the tables show only five. The first one (1:6) combines the unsupervised methods, i.e., every method but SVR, while the third (1:7) combines every method. Then, there is also a combination of the best two methods (2,5): TextRank and LSA; and the best three methods (2,4,5), that joins LexRank to the previous methods. Furthermore, there is a last experiment that combines only the methods that are based on graphs: TextRank, LexRank, and QueSTS.

For ROUGE 2, 3, 4, S and BERTScore, combining the unsupervised methods is the best approach, while for ROUGE 1, L, W and SU, the best approach is the combination of the graph methods (TextRank, LexRank, and QueSTS) and the combination of TextRank, LexRank and LSA. As for BLEURT, the best result is obtained with the combination of TextRank and LSA. All these combinations have TextRank in common, which is the method that obtained the best results individually.

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.346	0.175	0.233	0.080	0.039	0.052	0.027	0.013	0.018
TF-IDF Pre	**0.483**	0.210	0.293	0.017	0.008	0.011	0.000	0.000	0.000
TextRank HR	0.444	0.103	0.167	**0.143**	0.032	0.052	**0.056**	0.013	0.021
TextRank LR	0.323	0.209	0.255	0.080	0.050	**0.062**	0.026	0.017	0.021
TextRank Pre	0.442	0.247	0.317	0.016	0.009	0.012	0.000	0.000	0.000
QueSTS HR	0.206	0.306	0.246	0.040	0.061	0.048	0.012	0.018	0.014
QueSTS LR	0.085	**0.403**	0.140	0.013	0.071	0.022	0.003	0.018	0.005
QueSTS Pre	0.239	0.349	0.284	0.008	0.012	0.009	0.000	0.000	0.000
LexRank HR	0.438	0.104	0.168	0.139	0.032	0.052	0.053	0.013	0.020
Lexrank LR	0.313	0.213	0.253	0.074	0.050	0.060	0.023	0.017	0.019
LexRank Pre	0.420	0.250	0.314	0.014	0.009	0.011	0.000	0.000	0.000
LSA HR	0.435	0.113	0.179	0.130	0.032	0.052	0.048	0.012	0.019
LSA LR	0.270	0.250	0.260	0.050	0.048	0.049	0.015	0.016	0.015
LSA Pre	0.442	0.247	0.317	0.017	0.009	0.012	0.000	0.000	0.000
LC HR	0.190	0.312	0.236	0.051	**0.078**	**0.062**	0.019	**0.029**	**0.023**
LC LR	0.381	0.143	0.208	0.111	0.039	0.057	0.040	0.014	0.021
LC Pre	0.269	0.368	0.311	0.011	0.006	0.008	0.000	0.000	0.000
SVR HR	0.424	0.121	0.189	0.123	0.034	0.053	0.044	0.012	0.019
SVR LR	0.265	0.256	0.260	0.054	0.053	0.054	0.016	0.016	0.016
SVR Pre	0.349	0.306	**0.326**	0.011	0.010	0.010	0.000	0.000	0.000

Table 4.1: ROUGE 1, 2 and 3 for dataset PS5K. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.015	0.008	0.010	0.314	0.160	0.212	0.089	0.066	0.076
TF-IDF Pre	0.000	0.000	0.000	0.159	0.067	0.095	0.026	0.035	0.030
TextRank HR	**0.031**	0.008	**0.013**	**0.412**	0.096	0.155	**0.118**	0.039	0.059
TextRank LR	0.013	0.009	0.011	0.316	0.202	**0.247**	0.091	0.086	**0.088**
TextRank Pre	0.000	0.000	0.000	0.161	0.086	0.112	0.027	0.044	0.033
QueSTS HR	0.005	0.009	0.007	0.189	0.284	0.227	0.056	0.125	0.077
QueSTS LR	0.001	0.009	0.002	0.079	**0.379**	0.131	0.026	**0.189**	0.046
QueSTS Pre	0.000	0.000	0.000	0.092	0.141	0.111	0.016	0.080	0.027
LexRank HR	0.029	0.007	0.012	0.407	0.097	0.157	0.116	0.040	0.060
Lexrank LR	0.011	0.009	0.010	0.288	0.196	0.233	0.083	0.082	0.083
LexRank Pre	0.000	0.000	0.000	0.152	0.087	0.111	0.025	0.045	0.032
LSA HR	0.027	0.007	0.011	0.401	0.104	0.166	0.114	0.043	0.063
LSA LR	0.008	0.010	0.008	0.245	0.227	0.235	0.071	0.097	0.082
LSA Pre	0.000	0.000	0.000	0.161	0.087	0.113	0.027	0.045	0.033
LC HR	0.011	**0.017**	**0.013**	0.175	0.297	0.220	0.051	0.188	0.080
LC LR	0.022	0.008	0.011	0.351	0.133	0.193	0.100	0.062	0.077
LC Pre	0.000	0.000	0.000	0.101	0.236	0.141	0.017	0.172	0.031
SVR HR	0.024	0.007	0.011	0.389	0.112	0.174	0.111	0.046	0.065
SVR LR	0.008	0.008	0.008	0.240	0.233	0.237	0.069	0.099	0.082
SVR Pre	0.000	0.000	0.000	0.120	0.104	0.112	0.021	0.056	0.030

Table 4.2: ROUGE 4, L and W (weight 1.2) for dataset PS5K. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

When comparing the methods individually and combined, it is evident that there are no significant improvements in combining the methods. For ROUGE-1, the best individual F-Score is 0.260, while the combination of all the graph approaches (2:4) and the combination of the best three methods (2,4,5) reaches a value of 0.269, which although better, is not a significant difference. As for the other ROUGE scores, there is not any combination that outperforms the methods individually. However, they are able to reach very close values, with a maximum 0.002 difference between them. As for BERTScore, it presents

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
TF-IDF	0.071	0.035	0.047	0.117	0.059	0.078
TF-IDF Pre	0.055	0.024	0.034	0.127	0.055	0.077
TextRank HR	**0.126**	0.028	0.046	**0.179**	0.041	0.067
TextRank LR	0.071	0.046	**0.056**	0.114	0.073	**0.089**
TextRank Pre	0.053	0.030	0.039	0.118	0.066	0.085
QueSTS HR	0.037	0.057	0.045	0.065	0.098	0.079
QueSTS LR	0.013	**0.070**	0.022	0.025	**0.127**	0.042
QueSTS Pre	0.025	0.038	0.030	0.061	0.090	0.072
LexRank HR	0.122	0.028	0.046	0.175	0.041	0.067
Lexrank LR	0.066	0.045	0.054	0.108	0.073	0.087
LexRank Pre	0.049	0.030	0.037	0.112	0.067	0.084
LSA HR	0.113	0.028	0.045	0.167	0.042	0.068
LSA LR	0.044	0.042	0.043	0.082	0.077	0.079
LSA Pre	0.053	0.030	0.039	0.119	0.066	0.085
LC HR	0.046	0.067	0.054	0.070	0.114	0.087
LC LR	0.098	0.034	0.051	0.146	0.053	0.078
LC Pre	0.035	0.043	0.038	0.074	0.105	0.087
SVR HR	0.108	0.030	0.047	0.161	0.045	0.071
SVR LR	0.049	0.048	0.049	0.085	0.083	0.084
SVR Pre	0.035	0.032	0.033	0.087	0.078	0.082

Table 4.3: ROUGE S (skip-gram 4) and SU for dataset PS5K. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

Method	BERTScore			BLEURT
	R	P	F	
TF-IDF	-0.159	-0.305	-0.231	0.254
TF-IDF Pre	-0.110	-0.150	-0.129	0.170
TextRank HR	-0.106	-0.282	-0.194	0.272
TextRank LR	-0.117	-0.287	-0.202	0.287
TextRank Pre	-0.067	-0.155	**-0.110**	0.194
QueSTS HR	-0.144	-0.327	-0.236	0.268
QueSTS LR	-0.098	-0.395	-0.250	0.256
QueSTS Pre	-0.080	-0.221	-0.151	0.174
LexRank HR	-0.120	-0.302	-0.211	**0.289**
LexRank LR	-0.128	-0.308	-0.218	0.286
LexRank Pre	-0.069	-0.162	-0.115	0.189
LSA HR	-0.168	-0.297	-0.232	0.282
LSA LR	-0.150	-0.304	-0.227	0.272
LSA Pre	-0.069	-0.154	**-0.110**	0.194
LC HR	**0.008**	-0.402	-0.208	0.283
LC LR	-0.107	-0.314	-0.212	0.288
LC Pre	-0.119	-0.439	-0.286	0.176
SVR HR	-0.133	-0.296	-0.214	0.269
SVR LR	-0.168	-0.297	-0.232	0.247
SVR Pre	-0.091	**-0.143**	-0.116	0.171

Table 4.4: BERTScore and BLEURT in dataset PS5K. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

significantly worse values when combined. The best combined is -0.211, while individually the best is -0.110. Finally, BLEURT is also not able to reach the LexRank HR and LC LR results. These have values of 0.289 and 0.288, respectively, while when combined, BLEURT is only able to reach 0.287, with the combination of TextRank and LSA, which is pretty close but still lower.

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
2-1:6	0.351	0.182	0.240	0.090	0.045	**0.061**	**0.030**	0.016	**0.021**
2-2:4	0.270	0.267	**0.269**	0.059	0.059	0.059	0.018	0.018	0.018
3-1:7	0.224	**0.309**	0.260	0.047	0.064	0.054	0.015	0.021	0.018
1-2,5	**0.372**	0.165	0.228	0.017	**0.070**	0.027	0.005	**0.025**	0.009
2-2,4,5	0.276	0.262	**0.269**	**0.061**	0.057	0.059	0.019	0.018	0.018

Table 4.5: ROUGE 1, 2 and 3 for dataset PS5K with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
2-1:6	**0.016**	0.009	**0.011**	0.323	0.168	0.221	0.093	0.070	0.079
1-2:4	0.008	0.009	0.009	0.248	0.246	**0.247**	0.072	0.105	**0.086**
3-1:7	0.008	0.011	0.010	0.206	**0.285**	0.239	0.061	**0.126**	0.082
1-2,5	0.003	**0.015**	0.005	**0.343**	0.152	0.211	**0.098**	0.063	0.077
2-2,4,5	0.009	0.009	0.009	0.253	0.241	**0.247**	0.074	0.103	**0.086**

Table 4.6: ROUGE 4, L and W (weight 1.2) for dataset PS5K with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
2-1:6	0.080	0.041	**0.054**	**0.125**	0.064	0.085
1-2:4	0.053	0.053	0.053	0.089	0.089	**0.089**
3-1:7	0.043	**0.059**	0.049	0.073	0.101	0.085
1-2,5	**0.086**	0.038	0.053	0.029	**0.117**	0.047
2-2,4,5	0.054	0.052	0.053	0.091	0.087	**0.089**

Table 4.7: ROUGE S (skip-gram 4) and SU for dataset PS5K with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	BERTScore			BLEURT
	R	P	F	
2-1:6	**-0.122**	**-0.299**	**-0.211**	0.286
2-2:4	-0.133	-0.310	-0.222	0.281
3-1:7	-0.133	-0.314	-0.224	0.276
1-2,5	-0.252	-0.507	-0.382	**0.287**
2-2,4,5	-0.134	-0.309	-0.221	0.281

Table 4.8: BERTScore and BLEURT in dataset PS5K with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

4.4.2 Results in SciDuet

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.224	0.175	**0.196**	0.023	0.018	0.020	0.002	0.001	0.001
TF-IDF Pre	0.143	0.105	0.121	0.001	0.001	0.001	0.000	0.000	0.000
TextRank HR	0.295	0.102	0.152	**0.041**	0.014	0.021	**0.005**	0.002	**0.003**
TextRank LR	0.187	0.197	0.192	0.017	0.019	0.018	0.001	0.002	0.002
TextRank Pre	0.115	0.113	0.114	0.001	0.001	0.001	0.000	0.000	0.000
QueSTS HR	0.187	0.201	0.194	0.024	0.020	**0.022**	0.003	0.001	0.002
QueSTS LR	0.097	**0.275**	0.143	0.009	**0.023**	0.013	0.001	**0.003**	0.002
QueSTS Pre	0.110	0.110	0.110	0.001	0.001	0.001	0.000	0.000	0.000
LexRank HR	**0.297**	0.102	0.152	0.040	0.014	0.021	**0.005**	0.002	**0.003**
LexRank LR	0.196	0.194	0.195	0.019	0.020	0.019	0.002	0.002	0.002
LexRank Pre	0.112	0.108	0.110	0.001	0.001	0.001	0.000	0.000	0.000
LSA HR	0.296	0.114	0.164	0.036	0.013	0.019	0.003	0.001	0.002
LSA LR	0.164	0.212	0.185	0.011	0.016	0.013	0.001	0.001	0.001
LSA Pre	0.109	0.131	0.119	0.001	0.001	0.001	0.000	0.000	0.000
LC HR	0.174	0.191	0.182	0.021	0.020	0.020	0.003	**0.003**	**0.003**
LC LR	0.276	0.118	0.165	0.036	0.014	0.020	0.004	0.002	0.002
LC Pre	0.115	0.115	0.115	0.001	0.002	0.001	0.000	0.000	0.000
SVR HR	0.290	0.120	0.169	0.037	0.015	0.021	0.003	0.001	0.002
SVR LR	0.159	0.221	0.185	0.014	0.020	0.016	0.001	0.001	0.001
SVR Pre	0.090	0.134	0.108	0.000	0.001	0.001	0.000	0.000	0.000

Table 4.9: ROUGE 1, 2, 3 and 4 for dataset SciDuet. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.001	**0.001**	0.001	0.206	0.161	0.181	0.059	0.076	**0.066**
TF-IDF Pre	0.000	0.000	0.000	0.062	0.044	0.052	0.014	0.028	0.019
TextRank HR	0.002	**0.001**	0.001	0.274	0.095	0.141	0.076	0.043	0.055
TextRank LR	0.001	**0.001**	0.001	0.171	0.183	0.177	0.050	0.088	0.064
TextRank Pre	0.000	0.000	0.000	0.054	0.050	0.052	0.012	0.032	0.018
QueSTS HR	0.002	**0.001**	0.001	0.175	0.189	**0.182**	0.051	0.093	**0.066**
QueSTS LR	0.001	**0.001**	0.001	0.090	**0.259**	0.134	0.029	**0.142**	0.048
QueSTS HR Pre	0.000	0.000	0.000	0.051	0.053	0.052	0.012	0.035	0.017
LexRank HR	**0.003**	0.001	**0.002**	**0.276**	0.095	0.141	**0.077**	0.043	0.055
LexRank LR	0.001	**0.001**	0.001	0.182	0.181	0.181	0.053	0.087	**0.066**
LexRank Pre	0.000	0.000	0.000	0.051	0.047	0.049	0.012	0.030	0.017
LSA HR	0.001	**0.001**	0.001	0.273	0.105	0.151	0.076	0.048	0.059
LSA LR	0.000	**0.001**	0.000	0.149	0.194	0.168	0.044	0.094	0.060
LSA Pre	0.000	0.000	0.000	0.048	0.057	0.052	0.011	0.037	0.017
LC HR	0.002	**0.001**	0.001	0.161	0.179	0.170	0.045	0.106	0.064
LC LR	0.001	**0.001**	0.001	0.256	0.110	0.153	0.072	0.051	0.060
LC Pre	0.000	0.000	0.000	0.049	0.064	0.056	0.011	0.050	0.018
SVR HR	0.001	0.000	0.001	0.267	0.110	0.156	0.075	0.050	0.060
SVR LR	0.000	0.000	0.000	0.145	0.203	0.169	0.043	0.100	0.060
SVR Pre	0.000	0.000	0.000	0.041	0.061	0.049	0.010	0.041	0.016

Table 4.10: ROUGE L, W (weight 1.2), S (skip-gram 4) and SU for dataset SciDuet. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

Similar to PS5K, tests with lower ratios in SciDuet have, for the most part, achieved higher ROUGE scores (see Table 4.9, 4.10 and 4.11), with better precision and worse recall than experiments with higher ratios, with the exception of, once more, QueSTS and LC. The only difference lies in ROUGE-2, which has better F-scores with higher ratios for

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
TF-IDF	0.028	0.022	0.024	0.061	0.047	0.053
TF-IDF Pre	0.004	0.003	0.003	0.027	0.020	0.023
TextRank HR	**0.048**	0.017	0.025	0.089	0.031	0.046
TextRank LR	0.022	0.025	0.023	0.050	0.054	0.052
TextRank Pre	0.003	0.003	0.003	0.022	0.021	0.021
QueSTS HR	0.027	0.026	**0.027**	0.054	0.055	**0.055**
QueSTS LR	0.012	**0.032**	0.017	0.026	**0.074**	0.038
QueSTS HR Pre	0.003	0.003	0.003	0.021	0.021	0.021
LexRank HR	**0.048**	0.017	0.025	**0.090**	0.031	0.046
LexRank LR	0.024	0.025	0.025	0.053	0.053	0.053
LexRank Pre	0.021	0.020	0.020	0.021	0.020	0.020
LSA HR	0.043	0.016	0.024	0.086	0.033	0.047
LSA LR	0.015	0.020	0.017	0.040	0.053	0.045
LSA Pre	0.002	0.003	0.002	0.020	0.024	0.022
LC HR	0.025	0.025	0.025	0.050	0.055	0.052
LC LR	0.043	0.018	0.025	0.082	0.035	0.049
LC Pre	0.003	0.002	0.003	0.022	0.022	0.022
SVR HR	0.044	0.018	0.026	0.085	0.035	0.050
SVR LR	0.018	0.026	0.021	0.041	0.059	0.048
SVR Pre	0.002	0.003	0.002	0.017	0.025	0.020

Table 4.11: ROUGE S (skip-gram 4) and SU for dataset SciDuet. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

Method	BERTScore			BLEURT
	R	P	F	
TF-IDF	-0.268	-0.346	-0.306	0.203
TF-IDF Pre	-0.275	**-0.272**	-0.272	0.155
TextRank HR	-0.315	-0.398	-0.355	**0.224**
TextRank LR	-0.330	-0.395	-0.361	0.221
TextRank Pre	-0.335	-0.353	-0.342	0.178
QueSTS HR	-0.249	-0.362	-0.305	0.220
QueSTS LR	**-0.186**	-0.401	-0.295	0.212
QueSTS Pre	-0.263	-0.316	-0.288	0.166
LexRank HR	-0.265	-0.353	-0.308	0.221
LexRank LR	-0.277	-0.350	-0.312	0.222
LexRank Pre	-0.278	-0.293	-0.284	0.176
LSA HR	-0.255	-0.347	-0.300	0.214
LSA LR	-0.246	-0.353	-0.298	0.202
LSA Pre	-0.245	-0.281	**-0.262**	0.160
LC HR	-0.188	-0.395	-0.294	0.213
LC LR	-0.260	-0.354	-0.306	0.219
LC Pre	-0.276	-0.404	-0.342	0.163
SVR HR	-0.262	-0.345	-0.302	0.219
SVR LR	-0.285	-0.347	-0.315	0.196
SVR Pre	-0.248	-0.305	-0.275	0.151

Table 4.12: BERTScore and BLEURT in dataset SciDuet. LR corresponds to a lower ratio, and HR corresponds to a higher ratio. Pre corresponds to preprocessed texts.

every method. The behaviour of the methods when applied BERTScore and BLEURT (see Table 4.12) is also strikingly similar to that of the PS5K. For BERTScore, all methods and ratios display a higher recall than precision, except TF-IDF preprocessed, with every method except LSA and QueSTS having a better F-Score with a higher ratio. As for BLEURT, its performance is better, also with a higher ratio for every method except LexRank and Lexical Chains.

Applying preprocessing was only able to improve BERTScore, with the exception of the Lexical Chains method. Therefore, this approach does not bring much value to these experiements.

Furthermore, with this dataset, for ROUGE-1, the best method is TF-IDF, while for ROUGE-2, L, S, and SU, the best method is QueSTS HR. For ROUGE-4, the best is LexRank LR, and for ROUGE-3, the best is TextRank HR, LexRank HR, and LC HR. ROUGE-W has its best results with TF-IDF, QueSTS HR, and LexRank LR. So, in this dataset, the best methods are TF-IDF and QueSTS (their values are all close), followed by (in order): LexRank LR, TextRank LR, LC HR, and LSA LR. In terms of BERTScore, LSA with preprocessing (or LC HR without preprocessing) provides the best results. For BLEURT, the best method is TextRank HR, followed by LexRank and QueSTS HR.

It is challenging to identify a general best method for this dataset due to the diverse best-performing methods in the various metrics. However, looking at the best scores in related work using the same dataset, the best reported ROUGE-1, ROUGE-2, and ROUGE-L scores are respectively 0.20, 0.05, and 0.19. For ROUGE-1, simple unsupervised methods such as TF-IDF (0.196), LexRank LR (0.195) and QueSTS HR (0.194) perform very close to the state-of-the-art (0.20). For ROUGE-2, every method outperforms the best. For ROUGE-L, QueSTS HR (0.182), LexRank LR (0.181), and TF-IDF (0.181) are very close to the best obtained (0.19). Therefore, TF-IDF, LexRank LR, and QueSTS HR are the best methods to use for this dataset.

In an effort to improve the scores, some combinations of methods were also tested, as with the previous dataset PS5K. There were several combinations, but only six are shown in the tables 4.13, 4.14, 4.15 and 4.8. The first three are all combinations of the unsupervised methods with different thresholds (number of summaries a sentence needs to be in to be chosen for the final summary). Then, there is a combination of all the graph methods (2:4), all the methods (1:7), and only the best three methods (1,3,4): TF-IDF, QueSTS, and LexRank.

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
2-1:6	0.241	0.152	0.187	0.027	0.016	**0.020**	0.003	0.002	0.002
1-1:6	**0.321**	0.086	0.135	**0.046**	0.012	0.019	**0.007**	0.002	**0.003**
3-1:6	0.138	0.244	0.176	0.013	0.022	0.016	0.002	0.002	0.002
2-2:4	0.166	0.221	**0.189**	0.015	0.021	0.018	0.002	0.002	0.002
2-1,3,4	0.133	**0.246**	0.172	0.013	**0.023**	0.017	0.002	0.002	0.002
3-1:7	0.154	0.230	0.185	0.014	0.020	0.017	0.002	0.002	0.002

Table 4.13: ROUGE 1, 2, 3 and 4 in dataset SciDuet with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

For all combinations, several thresholds θ were tested to determine which was the best to use. The ideal θ is going to change depending on the number of summaries combined. The first three entries of the tables 4.13, 4.14, 4.15 and 4.8 show an example of the study of thresholds for the combination of the unsupervised methods. After that, for every combination, only the best θ was included.

For the combination of the unsupervised methods, the scores are best when the θ is 2, except for ROUGE 3 and 4, and BERTScore. A θ of 3 might be too restricting and one might be too embracing, which can even be seen by the recall and precision metrics.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
1-1:6	**0.004**	0.001	**0.002**	**0.298**	0.080	0.126	**0.083**	0.036	0.050
2-1:6	0.002	0.001	0.001	0.224	0.142	0.174	0.064	0.066	**0.065**
3-1:6	0.001	0.001	0.001	0.126	0.226	0.162	0.038	0.116	0.057
2-2:4	0.001	0.001	0.001	0.154	0.207	**0.176**	0.046	0.102	0.063
2-1,3,4	0.001	0.001	0.001	0.123	**0.228**	0.160	0.038	**0.117**	0.057
3-1:7	0.001	0.001	0.001	0.141	0.213	0.170	0.042	0.106	0.060

Table 4.14: ROUGE 4, L and W (weight 1.2) in dataset SciDuet with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
1-1:6	**0.055**	0.015	0.023	**0.100**	0.027	0.042
2-1:6	0.034	0.021	**0.026**	0.068	0.043	**0.053**
3-1:6	0.017	0.029	0.021	0.037	0.065	0.047
2-2:4	0.020	0.028	0.023	0.045	0.060	0.051
2-1,3,4	0.016	**0.030**	0.021	0.036	**0.066**	0.046
3-1:7	0.018	0.027	0.022	0.041	0.061	0.049

Table 4.15: ROUGE S (skip-gram 4) and SU in dataset SciDuet with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	BERTScore			BLEURT
	R	P	F	
1-1:6	-0.268	**-0.348**	-0.306	0.216
2-1:6	-0.259	-0.353	-0.305	0.216
3-1:6	-0.228	-0.374	**-0.301**	0.217
2-2:4	-0.266	-0.357	-0.310	**0.222**
2-1,3,4	**-0.223**	-0.382	-0.303	0.220
3-1:7	-0.246	-0.362	-0.303	0.214

Table 4.16: BERTScore and BLEURT in dataset SciDuet with methods combined. The first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Precision is the fraction of relevant sentences among the retrieved, while recall is the fraction of relevant sentences that are retrieved. With θ of 1, the recall is much higher than the precision, indicating that, even though many relevant sentences are included in the summary, because the summary is long, there are also many sentences that are not relevant. As for θ 3, the opposite happens, where the precision is higher than the recall, i.e., within the summary there are a good number of important sentences, but there are still many relevant sentences missing. Something similar happens in other combinations; if the θ is too high, it requires more sentences, and if the θ is too small, too many sentences are selected.

When comparing the results obtained by the methods, individually and combined, it

is evident that there is no benefit in combining the methods, such as with the previous dataset. It is preferable to apply the methods individually. For ROUGE-1 and L, the best combination is the graph methods, with their values being 0.189 and 0.176, respectively, which is lower than the best value obtained through the methods individually: 0.196 and 0.182. As for the other metrics, better scores are obtained when combining the unsupervised methods. BERTScore performs better with a θ of three, being able to obtain a value of-0.301 as opposed to the -0.262 obtained by LSA preprocessed. ROUGE 2, 3, 4, W, S, and SU perform better with a θ of two. These metrics obtain the following values: 0.020, 0.003, 0.002, 0.065, and 0.053, which are lower or equal to 0.22, 0.003, 0.002, 0.066, and 0.055 obtained by the methods individually.

4.5 Abstrative Methods

Abstractive methods do not extract full sentences as the extractive methods do, but instead generate a summary that captures the salient ideas of the source text, which may contain novel words and/or sentences. There are already trained ready-to-use models, referred to as transformers, that are available for use in this task.

As explained in section 4.1 these models can be fine-tuned for a specific dataset. However, the fine-tuned transformers that exist have not been fine-tuned with datasets relative to the same scope as those under study in this report, with most of them being composed of news and their summaries. Furthermore, the dataset records have short input texts and even shorter output texts, one or two sentences usually. Since the final objective is to construct slides, the final text needs to be longer than two sentences. A few highlights are not enough; some more detail is required.

Due to those limitations, an experiment where a transformer was fine-tuned to the PS5K and SciDuet datasets was carried out. This experiment, however, was not successful. When trying to implement fine-tuning for SciDuet and PS5K, the model could not process the entire length of the texts, making it necessary to compress the sentences to a small size. This leaves the dataset records with the same size as the ones used in the already fine-tuned transformers, and since almost everything in the text is cut, it is not possible to properly train the model. For that reason, fine-tuning a transformer with PS5K and SciDuet has been proven to not be a viable option, and therefore it was disregarded and the already fine-tuned transformers were used.

As a preliminary experiment, seven transformers were tested. These models are described in section 4.1. In order to test the transformers, the CNN/Daily Mail dataset [See et al., 2017] and [Hermann et al., 2015] was used, due to its reduced size.

Tables 4.17, 4.18 and 4.19 present the recall, precision, and F-score values for each transformer for ROUGE.

Transformer	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
T5 Large	0.106	0.270	0.152	0.029	0.077	0.042	0.012	0.034	0.018
Multilingual T5	0.132	0.249	0.173	0.028	0.054	0.036	0.009	0.017	0.012
Roberta2Roberta	0.151	0.249	0.188	0.022	0.038	0.028	0.005	0.008	0.006
Distilbart	0.300	**0.318**	0.309	0.111	**0.118**	0.114	0.061	**0.066**	0.064
Pegasus Wikihow	0.177	0.205	0.190	0.043	0.049	0.046	0.016	0.019	0.018
Pegasus CNN/Dailymail	**0.433**	0.272	**0.334**	**0.165**	0.103	**0.127**	**0.095**	0.059	**0.073**
Pegasus Xsum	0.115	0.217	0.151	0.029	0.054	0.038	0.012	0.023	0.015

Table 4.17: ROUGE 1, 2, 3 and 4 for various transformers in dataset CNN/Daily Mail.

Transformer	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
T5 Large	0.018	0.019	0.010	0.095	0.243	0.137	0.049	**0.210**	0.080
Multilingual T5	0.003	0.006	0.004	0.114	0.216	0.149	0.057	0.181	0.087
Roberta2Roberta	0.002	0.003	0.002	0.135	0.225	0.169	0.064	0.180	0.094
Distilbart	0.040	**0.043**	0.041	0.266	**0.282**	0.273	0.117	0.209	0.150
Pegasus Wikihow	0.008	0.009	0.008	0.151	0.176	0.163	0.073	0.143	0.097
Pegasus CNN/Dailymail	**0.063**	0.039	**0.048**	**0.375**	0.236	**0.289**	**0.170**	0.178	**0.174**
Pegasus Xsum	0.006	0.013	0.009	0.101	0.192	0.132	0.051	0.163	0.077

Table 4.18: ROUGE 4, L ans W (weight 1.2) for various transformers in dataset CNN/Daily Mail.

Transformer	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
T5 Large	0.018	0.054	0.027	0.033	0.098	0.050
Multilingual T5	0.017	0.036	0.023	0.038	0.079	0.052
Roberta2Roberta	0.017	0.030	0.021	0.036	0.062	0.045
Distilbart	0.083	**0.089**	0.086	0.118	**0.126**	0.122
Pegasus Wikihow	0.028	0.033	0.031	0.056	0.065	0.060
Pegasus CNN/Dailymail	**0.131**	0.080	**0.100**	**0.182**	0.111	**0.138**
Pegasus Xsum	0.019	0.038	0.026	0.037	0.077	0.050

Table 4.19: ROUGE S (skip-gram 4) and SU for various transformers in dataset CNN/Daily Mail.

When looking for the best transformers, it is clear that Distillbart and Pegasus CNN, particularly the latter, achieve the best results in all of the ROUGE metrics, with a significant gap in values between those two models and the other five. This may be due to the fact that both those datasets were trained with the same model as they were tested. In order to test PS5K and SciDuet, only the top two transformers were used.

Tables 4.20, 4.21, 4.22 and 4.23 present the ROUGE scores, BERTScore and BLEURT for the transformers, Distillbart and Pegasus CNN/Dailymail in the dataset PS5K. In these tables there are two versions of the mentioned transformers, one where the whole text is given and other with the text cut into pieces. The latter approach is used because the transformers will output a small summary, and so cutting the original text, summarize every piece separately, and then joining them together will result in a bigger final summary, which can be beneficial in a slide generation context.

The columns in the tables without the "T" correspond to the usual test where the original text component of the datasets is given as input, which will be then summarised by the transformers and the resulting output will be compared to the golden summary. Looking at the ROUGE scores, it is possible to observe that the F-Scores are very low, which is expected. The summaries that result from these transformers, as mentioned before, are only composed of one or two sentences, making the recall value really low. Recall stands for the quantity of relevant sentences that were retrieved, and since the recall is low, with no value surpassing 0.036, it is evident that the summaries are too short. On the other hand, the precision is quite high, with the highest value obtained being 0.499, which means that among the sentences retrieved there are a high percentage of relevant ones, which also makes sense, since there are few sentences, their value must be high. However, this still does not make up for the low recall, resulting in a low F-Score. So, in order to try to solve this problem, a new approach was created where the text is divided into topics and each topic is summarised individually before being combined into a final summary. So, for scientific papers, the several sections of the papers are taken as topics,

and each section is summarised separately. In the tables, these experiments are identified with a "T" for topic. The tables show that this method significantly raises the recall and F-score while lowering precision.

In table 4.23 the BERTScore and BLEURT are presented. Their behaviour is quite contradictory to the one resulting of the ROUGE scores. With these metrics the best performing methods are the ones without the text separated into topics. For BERTScore the recall is better for these methods and the precision is worse.

Transformer	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.031	**0.499**	0.059	0.006	**0.109**	0.011	0.002	**0.039**	0.004
Pegasus CNN T	0.266	0.258	**0.262**	0.053	0.051	0.052	0.014	0.014	0.014
Distilbart	0.036	0.484	0.066	0.007	0.104	0.013	0.002	0.036	0.004
Distilbart T	**0.316**	0.206	0.249	**0.070**	0.044	**0.054**	**0.019**	0.012	**0.015**

Table 4.20: ROUGE 1, 2, and 3 for transformers Distilbart and Pegasus CNN/DailyMail in dataset PS5K. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.001	**0.021**	0.002	0.029	**0.473**	0.055	0.011	**0.272**	0.022
Pegasus CNN T	0.006	0.006	0.006	0.242	0.236	**0.239**	0.070	0.101	**0.083**
Distilbart	0.001	0.020	0.002	0.033	0.454	0.062	0.012	0.256	0.024
Distilbart T	**0.009**	0.006	**0.007**	**0.287**	0.188	0.227	**0.083**	0.079	0.081

Table 4.21: ROUGE 4, L and W (weight 1.2) for transformers Distilbart and Pegasus CNN/DailyMail in dataset PS5K. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
Pegasus CNN	0.005	**0.097**	0.009	0.009	**0.168**	0.018
Pegasus CNN T	0.047	0.046	0.047	0.084	0.082	**0.083**
Distilbart	0.005	0.090	0.010	0.010	0.159	0.020
Distilbart T	**0.062**	0.039	**0.048**	**0.104**	0.067	0.082

Table 4.22: S (skip-gram 4) and SU for transformers Distilbart and Pegasus CNN/DailyMail in dataset PS5K. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	BERTScore			BLEURT
	R	P	F	
Pegasus CNN	**0.053**	-0.445	**-0.208**	**0.247**
Pegasus CNN T	-0.181	-0.314	-0.247	0.244
Distilbart	0.030	-0.438	-0.214	0.245
Distilbart T	-0.195	**-0.311**	-0.252	0.239

Table 4.23: BERTScore and BLEURT for transformers Distilbart and Pegasus CNN/DailyMail in dataset PS5K. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.049	**0.489**	0.088	0.011	**0.127**	0.020	0.004	**0.054**	0.007
Pegasus CNN T	0.282	0.339	0.308	0.067	0.085	0.075	0.025	0.035	**0.029**
Distilbart	0.070	0.464	0.122	0.016	0.109	0.028	0.006	0.043	0.010
Distilbart T	**0.337**	0.288	**0.310**	**0.084**	0.073	**0.078**	**0.031**	0.028	**0.029**

Table 4.24: ROUGE 1, 2, 3 and 4 for transformers Distilbart and Pegasus CNN/DailyMail in dataset PS5K. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.002	**0.033**	0.004	0.045	**0.458**	0.082	0.016	**0.271**	0.030
Pegasus CNN T	0.015	0.022	**0.018**	0.252	0.305	**0.276**	0.074	0.146	0.098
Distilbart	0.003	0.024	0.005	0.064	0.425	0.111	0.022	0.240	0.040
Distilbart T	**0.018**	0.017	0.017	**0.299**	0.256	**0.276**	**0.086**	0.120	**0.100**

Table 4.25: ROUGE 4, L and W (weight 1.2) for transformers Distilbart and Pegasus CNN/DailyMail in dataset SciDuet. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

The same process was used for the SciDuet dataset. The tables 4.24, 4.25 and 4.26 present the ROUGE scores, which have very similar behaviour to the results in the previous dataset. When the entire original text is presented, recall and F-Score are low and precision is high, whereas when the text is divided into topics, recall and F-Score are higher.

As for BERTScore and BLEURT, as is shown in table 4.27, they present their best final scores when the text is divided by topics, even though the gap between the methods results is not big. BERTScore has better recall and worse precision when the text is complete, which is contrary to what happens in ROUGE.

In the following section, the differences between the two types of summarization—extractive and abstractive—as well as the key findings—are highlighted.

4.6 Discussion

It is not an easy task to determine which is the best method after the experiments presented in the sections above. This happens because some methods do better than others in certain metrics, and vice versa, and it is not possible to be certain which one is more relevant.

Transformer	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
Pegasus CNN	0.009	**0.105**	0.016	0.015	**0.173**	0.028
Pegasus CNN T	0.059	0.076	0.067	0.097	0.120	0.107
Distilbart	0.012	0.093	0.022	0.022	0.157	0.039
Distilbart T	**0.072**	0.065	**0.068**	**0.117**	0.102	**0.109**

Table 4.26: ROUGE S (skip-gram 4) and SU for transformers Distilbart and Pegasus CNN/DailyMail in dataset SciDuet. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

Transformer	BERTScore			BLEURT
	R	P	F	
Pegasus CNN	**0.061**	-0.415	-0.188	0.217
Pegasus CNN T	-0.130	-0.220	-0.174	0.217
Distilbart	0.020	-0.373	-0.184	0.221
Distilbart T	-0.138	**-0.199**	**-0.167**	**0.227**

Table 4.27: BERTScore and BLEURT for transformers Distilbart and Pegasus CNN/DailyMail in dataset SciDuet. "T" stands for Topic, indicating that the text has been divided into topics, each of which has been summarised separately, before being combined into a final summary.

There are two types of methods: extractive or abstractive. Extractive methods are simple methods that extract sentences exactly as they are from the original text without adding any novelty to them or condense sentences that have much unnecessary information. Most of the methods studied here are unsupervised, which is an important advantage since they do not require any training data, making it possible to summarise every text in every topic and language. As for the abstractive methods, they aim to build a summary more like a human would, adding new words or sentences. Typically, these methods output very concise summaries that give a brief highlight of what the original text is about. The abstractive methods in this report are all transformers, i.e., supervised models that are already trained and ready to use.

The sections above 4.4 and 4.5 present the results of the datasets PS5K and SciDuet for all the extractive and abstractive methods. In PS5K, the abstractive methods perform worse than the extractive methods, while in SciDuet the opposite happens.

For the dataset PS5K, the tables 4.1, 4.2, 4.3 and 4.4 display the outcomes of extractive methods, while the tables 4.20, 4.21, 4.22 and 4.23 display the outcomes of abstractive methods. As one can observe, every score of the extractive methods is higher than the scores of the abstractive methods, with the exception of ROUGE-1, which, without preprocessing, achieves a highest result of 0.260, while the transformer Pegasus CNN/DailyMail, when the text is divided by topics, is able to reach 0.262. However, this is only one metric among ten. So, for this dataset, it is preferable to use an extractive method, such as TextRank, that has always better values and is unsupervised. Following that method is: LSA LR, LexRank LR, QueSTS HR, LC HR, and TF-IDF. All these methods have close scores. As for the abstractive methods, both transformers, Pegasus CNN/DailyMail and Distillbart, present very close scores, with the Distillbart having an advantage in BERTScore and BLEURT. The results are better when the text is divided by topics.

As for SciDuet, the tables 4.9, 4.10, 4.11 and 4.12 display the outcomes of extractive methods, while the tables 4.24, 4.25, 4.26 and 4.27 display the outcomes of abstractive methods. Contrarily to PS5K, all metrics have better values for the abstractive methods, with the Distillbart having the best results. The best extractive methods are TF-IDF and QueSTS (their scores are all close), followed by LexRank LR, TextRank LR, LC HR, and LSA LR. The differences between the methods can be quite significant, with, for example, ROUGE-1 having a difference of 0.114 between the best abstractive and extractive methods.

The dissimilarity between the two datasets makes it challenging to determine which approach is best: extractive or abstractive. The methods to choose are going to depend on the dataset to be used. However, abstractive methods have the disadvantage of needing training data, which can be of low quality, hard to access or even inexistent. Since training is not required, unsupervised extractive methods allow for a quicker summarization of any

text with any topic or language. Furthermore, abstractive methods tend to provide smaller, more concise summaries, which for slide generation may not be ideal. Even though breaking the summary into topics helps with this problem, the generated summaries are still not going to be size adjustable as most extractive methods are, i.e., it will not be possible to determine a ratio of the original text that is going to be kept in the summary. In the case of wanting to generate larger abstractive summaries, the original text would have to be separated into smaller pieces, which might still not return the summary with the intended size and could create new problems and decrease the quality. Forcing a part of the text to be in the slide might not be ideal since that part might not be needed and could decrease the general quality of the summary. Additionally, there are other factors, such as coherence, that the automatic metrics cannot evaluate but that might be important, especially for transformers, that in order to create abstractive summaries might "hallucinate" and start to invent facts that are not true. This does not happen for extractive methods because they do not create any new words or sentences; they simply stick to what is written in the original text.

Chapter 5

Slide Generation

The previous chapter 4 presents the results for the automatic evaluation of several summarization methods for two datasets: SciDuet and PS5K. With the summaries made, the next step is to put those summaries into slide decks. After that, a set of people are going to evaluate them. What might be good in terms of summarization might not apply to slides. Aspects like information organisation are something that is only possible to evaluate by giving the slides.

However, as explained in section 4.2, the datasets SciDuet and PS5K are very challenging to understand. They are composed of scientific papers with abundant complex concepts and mathematical formulas. So, in order to evaluate their summary or slide decks, it would be required to have people that are experts in the subjects, which is something very hard to accomplish. Therefore, it is more viable to have a dataset with simple, easy-to-understand concepts that people may already have some knowledge about in order to accelerate the evaluation process. This is a very important aspect since manually evaluating slide decks/summaries is a process that demands much time and effort. This evaluation requires every evaluator to read the original document and the generated slides or summary and then compare them and answer the provided questions. If the documents are too long and hard to understand, this will be an intense task, and no one will want to participate, and even if they do, they may not understand the topic well and their evaluation may not be reliable. For these reasons, a dataset should be composed of topics that are widely known and easy to comprehend. Additionally, this type of data is more akin to that of Mindflow (a project-related company), where the slides concern a presentation of a theme. However, datasets are not only composed of text; they also need their respective summaries, which can be harder to encounter. So, with that in mind, a dataset with Wikipedia articles was built.

This chapter is divided into four sections. The first one (5.1) explains how the Wikipedia dataset was built; section 5.2 presents the automatic evaluation, i.e., the results of the metrics ROUGE, BERTScore, and BLEURT when applied to the dataset Wikipedia in both English and Portuguese. After that, the construction of slide decks is explained in section 5.3, along with some examples. Finally, in section 5.4 the process for the human evaluation is described and the outcomes are shown and discussed.

5.1 Wikipedia Dataset

Wikipedia has a wide range of topics that one may choose from, and at the beginning of each there is a small introduction that can be taken as a summary. Since we could not find such a dataset, one had to be constructed. For that, ten different articles were chosen. These were divided into six different categories, and for each one a maximum of two articles were chosen. The articles and categories are: Places ("Coimbra" and "Europe"), Organization/Band ("University of Coimbra" and "Queen"), Events ("Carnation Revolution"), Technical Article ("Pythagorean theorem" and "Programming Language"), Person ("Cristiano Ronaldo" and "Luís de Camões") and Movie ("Star Wars"). Wikipedia also allows for the construction of datasets in several languages, which is an advantage compared to PS5K and SciDuet, since they are only composed of English articles. For the experiments taking place here, two datasets were built: one in English and one in Portuguese, since the objective was to build the slide decks in both of those languages. The experiments so far only took English into account because there are no datasets for slide generation in Portuguese. Despite being a small summarization dataset, this will allow for a preliminary experiment to see how Portuguese-language slides would appear. Below there is an example of the Wikipedia article "Carnation Revolution" (abbreviated), and its corresponding summary.

Article:
By the 1970s, nearly a half-century of authoritarian rule weighed on Portugal. The 28 May 1926 coup d'état implemented an authoritarian regime incorporating social Catholicism and integralism. In 1933, the regime was renamed Estado Novo (New State). António de Oliveira Salazar served as Prime Minister until 1968.In sham elections the government candidate usually ran unopposed, while the opposition used the limited political freedoms allowed during the brief election period to protest, withdrawing their candidates before the election to deny the regime political legitimacy. The Estado Novo's political police, the PIDE (Polícia Internacional e de Defesa do Estado, later the DGS, Direcção-Geral de Segurança and originally the PVDE, Polícia de Vigilância e Defesa do Estado), persecuted opponents of the regime, who were often tortured, imprisoned or killed.In 1958, General Humberto Delgado, a former member of the regime, stood against the regime's presidential candidate, Américo Tomás, and refused to allow his name to be withdrawn. Tomás won the election amidst claims of widespread electoral fraud, and the Salazar government abandoned the practice of popularly electing the president and gave the task to the National Assembly.Portugal's Estado Novo government remained neutral in the second world war, and was initially tolerated by its NATO post-war partners due to its anti-communist stance.
...
In February 1974, Caetano decided to remove General António de Spínola from the command of Portuguese forces in Guinea in the face of Spínola's increasing disagreement with the promotion of military officers and the direction of Portuguese colonial policy. This occurred shortly after the publication of Spínola's book, Portugal and the Future, which expressed his political and military views of the Portuguese Colonial War. Several military officers who opposed the war formed the MFA to overthrow the government in a military coup. The MFA was headed by Vítor Alves, Otelo Saraiva de Carvalho and Vasco Lourenço, and was joined later by Salgueiro Maia. The movement was aided by other Portuguese army officers who supported Spínola and democratic civil and military reform. It is speculated that Francisco da Costa Gomes actually led the revolution.The coup had two secret signals. First, Paulo de Carvalho's "E Depois do Adeus" (Portugal's entry in the 1974 Eurovision Song Contest) was aired on Emissores Associados de Lisboa at 10:55 p.m. on 24 April.

This alerted rebel captains and soldiers to begin the coup. The second signal came at 12:20 a.m. on 25 April, when Rádio Renascença broadcast "Grândola, Vila Morena" (a song by Zeca Afonso, an influential political folk musician and singer who was banned from Portuguese radio at the time). The MFA gave the signals to take over strategic points of power in the country. Six hours later, the Caetano government relented.
...
After an early period of turmoil, Portugal emerged as a democratic country. The country divested itself of almost all of its former colonies and experienced severe economic turmoil. For the Portuguese and their former colonies this was a very difficult period, but civil rights and political freedoms were achieved. Construction of what is now called the 25 de Abril Bridge began on 5 November 1962. It opened on 6 August 1966 as the Salazar Bridge, named after Estado Novo leader António de Oliveira Salazar. Soon after the Carnation Revolution of 1974, the bridge was renamed the 25 de Abril Bridge to commemorate the revolution. Citizens who removed the large, brass "Salazar" sign from a main pillar of the bridge and painting a provisional "25 de Abril" in its place were recorded on film. Many Portuguese streets and squares are named vinte e cinco de Abril (25 April), for the day of the revolution. The Portuguese Mint chose the 40th anniversary of the Carnation Revolution for its 2014 2 euro commemorative coin. Freedom Day (25 April) is a national holiday, with state-sponsored and spontaneous commemorations of the civil liberties and political freedoms achieved after the revolution.
...

Summary:
The Carnation Revolution (Portuguese: Revolução dos Cravos), also known as the 25 April (Portuguese: 25 de Abril), was a military coup by left-leaning military officers that overthrew the authoritarian Estado Novo regime on 25 April 1974 in Lisbon, producing major social, economic, territorial, demographic, and political changes in Portugal and its overseas colonies through the Processo Revolucionário Em Curso. It resulted in the Portuguese transition to democracy and the end of the Portuguese Colonial War. The revolution began as a coup organised by the Armed Forces Movement (Portuguese: Movimento das Forças Armadas, MFA), composed of military officers who opposed the regime, but it was soon coupled with an unanticipated, popular civil resistance campaign. Negotiations with African independence movements began, and by the end of 1974, Portuguese troops were withdrawn from Portuguese Guinea, which became a UN member state. This was followed in 1975 by the independence of Cape Verde, Mozambique, São Tomé and Príncipe and Angola in Africa and the declaration of independence of East Timor in Southeast Asia. These events prompted a mass exodus of Portuguese citizens from Portugal's African territories (mostly from Angola and Mozambique), creating over a million Portuguese refugees – the retornados.The carnation revolution got its name from the fact that almost no shots were fired and from restaurant worker Celeste Caeiro offering carnations to the soldiers when the population took to the streets to celebrate the end of the dictatorship, with other demonstrators following suit and carnations placed in the muzzles of guns and on the soldiers' uniforms. In Portugal, 25 April is a national holiday (Portuguese: Dia da Liberdade, Freedom Day) that commemorates the revolution.

5.2 Automatic Evaluation

This section presents the results of the metrics ROUGE, BERTScore, and BLEURT when applied to the dataset Wikipedia in both English (subsection 5.2.1) and Portuguese (subsection 5.2.2).

5.2.1 English

Tables 5.1, 5.2, 5.3 and 5.4 present the ROUGE, BERTScore and BLEURT results when applied to the extractive methods and their combinations.

ROUGE-1 and BERTScore have better results with the combination of the unsupervised methods with a threshold three, while ROUGE 2, 3, 4, W, S and SU have a better performance with Lexical Chains. As for ROUGE L the best method is QueSTS. This method is also the best for ROUGE 1 if the combinations were excluded.

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.624	0.111	0.189	0.189	0.033	0.056	0.052	0.009	0.015
TextRank LR	0.609	0.108	0.184	0.199	0.035	0.059	0.063	0.010	0.018
QueSTS HR	0.440	**0.238**	0.309	0.118	0.056	0.076	0.034	0.010	0.015
LexRank LR	0.574	0.144	0.230	0.177	0.042	0.067	0.056	0.012	0.019
LSA LR	0.485	0.208	0.291	0.097	0.039	0.056	0.023	0.009	0.013
LC HR	0.614	0.151	0.243	0.219	**0.075**	**0.111**	0.072	**0.039**	**0.051**
3-1:6	0.485	0.232	**0.314**	0.136	0.064	0.087	0.040	0.017	0.024
2-1:6	**0.663**	0.082	0.147	**0.232**	0.029	0.051	**0.078**	0.009	0.017
1-3,5	0.578	0.141	0.227	0.158	0.034	0.055	0.045	0.007	0.012

Table 5.1: ROUGE 1, 2 and 3 in dataset Wikipedia in English for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.023	0.004	0.006	0.395	0.070	0.118	0.079	0.035	0.049
TextRank LR	0.032	0.005	0.009	0.393	0.069	0.117	0.080	0.035	0.048
QueSTS HR	0.015	0.003	0.005	0.278	**0.155**	**0.199**	0.059	**0.083**	0.069
LexRank LR	0.024	0.004	0.007	0.363	0.089	0.143	0.074	0.045	0.056
LSA LR	0.006	0.002	0.003	0.287	0.121	0.170	0.060	0.063	0.061
LC HR	0.028	**0.025**	**0.026**	0.415	0.110	0.174	0.083	0.062	**0.071**
3-1:6	0.017	0.007	0.010	0.300	0.142	0.193	0.062	0.072	0.066
2-1:6	**0.033**	0.004	0.007	**0.432**	0.053	0.095	**0.087**	0.027	0.041
1-3,5	0.018	0.002	0.004	0.365	0.087	0.141	0.074	0.044	0.055

Table 5.2: ROUGE 4, L and W (weight 1.2) in dataset Wikipedia in English for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

As for the abstractive methods, these are presented in tables 5.5, 5.6, 5.7 and 5.8. As expected, not separating the texts into topics results in lower scores. However, these scores are higher than the ones obtained from the other datasets: PS5K and SciDuet. This happens because the summaries taken from Wikipedia are smaller, making the fraction of relevant sentences that were retrieved (recall) higher, which results in a better F-Score. This is also relevant for other metrics that have higher scores. For example, ROUGE-1 and BERTScore. From all the transformers, the best performing one is Distillbart, which is in line with the results of the other datasets that also presented this model as the best.

It is clear that the abstractive methods outperform the extractive ones when compared

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
TF-IDF	0.192	0.033	0.057	0.265	0.046	0.079
TextRank LR	0.192	0.034	0.057	0.263	0.046	0.078
QueSTS HR	0.113	0.059	0.077	0.168	**0.089**	0.116
LexRank LR	0.169	0.040	0.065	0.237	0.058	0.093
LSA LR	0.093	0.038	0.054	0.159	0.066	0.093
LC HR	0.214	**0.065**	**0.100**	0.281	0.080	**0.125**
3-1:6	0.129	0.061	0.083	0.189	**0.089**	0.121
2-1:6	**0.226**	0.028	0.050	**0.300**	0.037	0.066
1-3,5	0.155	0.034	0.056	0.226	0.052	0.085

Table 5.3: ROUGE S (skip-gram 4) and SU in dataset Wikipedia in English for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	BERTScore			BLEURT
	R	P	F	
TF-IDF	-0.086	-0.099	-0.092	0.293
TextRank LR	-0.097	-0.099	-0.097	0.307
QueSTS HR	-0.055	-0.125	-0.089	0.315
LexRank LR	-0.075	-0.086	-0.080	0.307
LSA LR	-0.111	-0.116	-0.113	0.291
LC HR	**-0.040**	-0.114	-0.080	0.320
3-1:6	-0.063	**-0.073**	**-0.067**	**0.327**
2-1:6	-0.059	-0.088	-0.073	0.325
1-3,5	-0.097	-0.109	-0.102	0.290

Table 5.4: BERTScore and BLEURT in dataset Wikipedia in English for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

to each other. The size of the summary might have some impact on these scores. The extractives are larger, whereas the transformers return a summary that is smaller and more akin to the ones from the golden summary.

Transformer	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.077	**0.461**	0.132	0.024	**0.136**	0.041	0.009	**0.048**	0.016
Pegasus CNN T	0.365	0.300	0.329	0.090	0.077	0.083	0.023	0.021	0.022
Distillbart	0.106	0.420	0.169	0.021	0.085	0.033	0.006	0.021	0.009
Distillbart T	**0.438**	0.283	**0.344**	**0.113**	0.071	**0.087**	**0.030**	0.018	**0.023**

Table 5.5: ROUGE 1, 2 and 3 in dataset Wikipedia in English for the abstractive methods.

5.2.2 Portuguese

Tables 5.9, 5.10, 5.11 and 5.12 present the ROUGE, BERTScore and BLEURT scores for the Portuguese Wikipedia articles. The best performing methods for this dataset are LexRank, which has the best results in ROUGE 2, 3, 4, W, S, and SU; QueSTS, which excels in BERTScore; TextRank, which is better for BLEURT; and LSA, which outperforms

Transformer	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
Pegasus CNN	0.005	**0.026**	0.009	0.061	**0.366**	0.104	0.016	**0.240**	0.029
Pegasus CNN T	0.010	0.009	**0.010**	0.231	0.192	0.209	0.049	0.102	0.066
Distillbart	0.002	0.008	0.003	0.078	0.318	0.126	0.020	0.200	0.036
Distillbart T	**0.012**	0.007	0.008	**0.271**	0.174	**0.212**	**0.057**	0.091	**0.070**

Table 5.6: ROUGE 4, L and W (weight 1.2) in dataset Wikipedia in English for the abstractive methods.

Transformer	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
Pegasus CNN	0.019	**0.112**	0.032	0.029	**0.175**	0.049
Pegasus CNN T	0.080	0.068	0.073	0.128	0.107	0.116
Distillbart	0.018	0.075	0.028	0.032	0.135	0.052
Distillbart T	**0.102**	0.064	**0.079**	**0.158**	0.101	**0.123**

Table 5.7: ROUGE S (skip-gram 4) and SU in dataset Wikipedia in English for the abstractive methods.

Transformer	BERTScore			BLEURT
	R	P	F	
Pegasus CNN	**0.061**	-0.317	-0.135	**0.296**
Pegasus CNN T	-0.086	-0.063	-0.074	0.264
Distillbart	-0.020	-0.289	-0.157	0.282
Distillbart T	-0.091	**-0.050**	**-0.070**	0.263

Table 5.8: BERTScore and BLEURT in dataset Wikipedia in English for the abstractive methods.

all others in ROUGE 1 and L, even though LexRank is not far behind from its results. Furthermore, if combinations are employed, they are able to outperform some methods. This is true for ROUGE-1 and L, which perform better when the unsupervised methods are combined with a threshold of three and when QueSTS, LexRank, and LSA are combined with a threshold of two. Among the two combinations, the latter is preferable.

In contrast to the results of the previous datasets, the majority of BERTScore values are positive, while BLEURT values are negative. This might be happening simply because of the different languages used. Although these metrics can be applied to a variety of languages, it is reasonable to assume that since English is a more popular language, it has received better training and preparation. As a result, the accuracy in the Portuguese language result might not be as good, making the BERTScore look better and the BLEURT look worse. On the other hand, ROUGE has range values that are comparable to those in the prior datasets, which makes sense since this metric only compares words exactly as they are in text, without needing any training, contrarily to the other evaluation metrics.

Tables 5.13, 5.14, 5.15 and 5.16 present the results for the transformer Multilingual T5. Due to there only being one, the only comparison to make is between the input text being separated by topics or not, and, as is expected, separating the text brings a big improvement to the method. Furthermore, it is clear from a comparison of this method and extractive methods that the latter performs superior to the former. This is contrary to what happens when the Wikipedia dataset is in English. The reason for this is simply that different transformers are used, and because this one was trained for several languages, its results are not as accurate as Distillbart or Pegasus CNN. Tables 4.17, 4.18 and 4.19 show the various differences in the metrics' results.

Method	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.516	0.179	0.266	0.128	0.040	0.062	0.031	0.011	0.016
TextRank LR	0.558	0.186	0.278	0.154	0.048	0.074	0.047	0.016	0.023
QueSTS HR	0.541	0.207	0.299	0.129	0.049	0.071	0.034	0.015	0.021
LexRank LR	0.514	0.225	0.313	0.140	0.056	**0.080**	0.044	0.018	**0.025**
LSA LR	0.421	0.276	0.333	0.076	0.047	0.058	0.017	0.011	0.014
LC HR	0.134	**0.466**	0.208	0.030	**0.141**	0.050	0.010	**0.058**	0.017
2-1:6	0.584	0.172	0.266	0.161	0.047	0.073	0.047	0.015	0.023
3-1:6	0.305	0.382	0.339	0.062	0.081	0.070	0.015	0.021	0.017
1-4,5	0.583	0.166	0.258	0.173	0.045	0.072	0.056	0.014	0.023
1-3:5	**0.654**	0.118	0.201	**0.215**	0.037	0.064	**0.069**	0.012	0.021
2-3:5	0.405	0.331	**0.364**	0.077	0.061	0.068	0.017	0.015	0.016

Table 5.9: ROUGE 1, 2 and 3 in dataset Wikipedia in Portuguese for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
TF-IDF	0.007	0.002	0.003	0.299	0.105	0.155	0.054	0.053	0.053
TextRank LR	0.015	0.005	0.007	0.319	0.107	0.160	0.058	0.054	0.056
QueSTS HR	0.010	0.005	0.006	0.310	0.126	0.179	0.056	0.064	**0.060**
LexRank LR	0.016	0.006	**0.009**	0.299	0.130	0.181	0.056	0.066	**0.060**
LSA LR	0.007	0.004	0.005	0.231	0.155	0.185	0.043	0.080	0.056
LC HR	0.003	**0.024**	0.006	0.095	**0.367**	0.151	0.021	**0.245**	0.038
2-1:6	0.015	0.005	0.007	0.336	0.101	0.155	0.061	0.050	0.055
3-1:6	0.004	0.005	0.004	0.177	0.229	0.200	0.033	0.128	0.052
1-4,5	0.022	0.005	0.008	0.343	0.096	0.151	0.063	0.047	0.054
1-3:5	**0.027**	0.005	0.008	**0.389**	0.071	0.120	**0.070**	0.035	0.047
2-3:5	0.004	0.004	0.004	0.224	0.189	**0.205**	0.041	0.099	0.058

Table 5.10: ROUGE 4, L and W (weight 1.2) in dataset Wikipedia in Portuguese for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

5.3 Slide Generation

To complete the pipeline of slide generation, the sentences of the generated summaries need to be organised into slides. While a more complex strategy could be devised [Sefid et al., 2021b, Sun et al., 2021b], and will be in the future, a fairly simple one was adopted for illustrating the process in this report. Having in mind that most textual documents have sections, their titles were used as the topics for the slides. The sentences in the summary are then grouped according to those topics, i.e., each sentence is associated with the topic (section) under which it was placed in the original document. As a result, each slide will feature a title (i.e., the title of the section) followed by sentences selected from the original document, in their original order. Each slide can only contain a configurable number of sentences or characters. If the text does not fit on a single slide, additional text-only slides are made. For illustrative purposes, Appendix A show some examples of the slides generated with QueSTS HR, Distillbart, and TextRank LR.

Method	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
TF-IDF	0.128	0.041	0.062	0.193	0.064	0.096
TextRank LR	0.148	0.047	0.071	0.217	0.070	0.106
QueSTS HR	0.140	0.050	0.074	0.207	0.077	0.112
LexRank LR	0.136	0.055	**0.079**	0.199	0.084	0.118
LSA LR	0.076	0.048	0.059	0.134	0.087	0.105
LC HR	0.030	**0.114**	0.048	0.048	**0.179**	0.075
2-1:6	0.158	0.046	0.071	0.229	0.067	0.103
3-1:6	0.063	0.082	0.071	0.103	0.134	0.117
1-4,5	0.167	0.044	0.070	0.237	0.065	0.101
1-3:5	**0.206**	0.036	0.061	**0.282**	0.050	0.085
2-3:5	0.082	0.066	0.073	0.136	0.111	**0.122**

Table 5.11: ROUGE S (skip-gram 4) and SU in dataset Wikipedia in Portuguese for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Method	BERTScore			BLEURT
	R	P	F	
TF-IDF	0.127	0.159	0.144	-0.134
TextRank LR	0.140	0.177	0.159	**-0.101**
QueSTS HR	**0.196**	**0.193**	**0.195**	-0.122
LexRank LR	-0.101	-0.112	-0.105	-0.126
LSA LR	0.117	0.144	0.131	-0.131
LC HR	0.164	-0.047	0.050	-0.107
2-1:6	0.166	0.177	0.172	-0.122
3-1:6	0.173	0.137	0.153	-0.112
1-4,5	0.153	0.171	0.163	-0.123
1-3:5	0.170	0.183	0.177	-0.127
2-3:5	**0.196**	0.185	0.192	-0.111

Table 5.12: BERTScore and BLEURT in dataset Wikipedia in Portuguese for both separate and combined extractive methods. In the combinations the first number represents the number of summaries a sentence needs to be to be chosen to the final summary. The final numbers represent the methods combined. For each method is given a number. TF-IDF - 1, TextRank - 2, QueSTS - 3, LexRank - 4, LSA - 5, Lexical Chains - 6 and SVR - 7.

Transformer	ROUGE-1			ROUGE-2			ROUGE-3		
	R	P	F	R	P	F	R	P	F
Multilingual	0.025	**0.718**	0.048	0.011	**0.342**	0.021	0.005	**0.161**	0.010
Multilingual T	**0.148**	0.578	**0.236**	**0.040**	0.161	**0.064**	**0.015**	0.053	**0.023**

Table 5.13: ROUGE 1, 2 and 3 in dataset Wikipedia in Portuguese for the abstractive methods.

Transformer	ROUGE-4			ROUGE-L			ROUGE-W		
	R	P	F	R	P	F	R	P	F
Multilingual	0.003	**0.086**	0.006	0.022	**0.644**	0.043	0.006	**0.468**	0.012
Multilingual T	**0.005**	0.016	**0.008**	**0.109**	0.432	**0.174**	**0.024**	0.251	**0.044**

Table 5.14: ROUGE 4, L and W (weight 1.2) in dataset Wikipedia in Portuguese for the abstractive methods.

Transformer	ROUGE-S			ROUGE-SU		
	R	P	F	R	P	F
Multilingual	0.008	**0.295**	0.016	0.011	**0.388**	0.021
Multilingual T	**0.036**	0.148	**0.058**	**0.055**	0.221	**0.088**

Table 5.15: ROUGE S (skip-gram 4) and SU in dataset Wikipedia in Portuguese for the abstractive methods.

Transformer	BERTScore			BLEURT
	R	P	F	
Multilingual	**0.213**	-0.250	-0.050	-0.083
Multilingual T	0.184	**-0.046**	**0.062**	**-0.052**

Table 5.16: BERTScore and BLEURT in dataset Wikipedia in Portuguese for the abstractive methods.

It is important to remember that these slide decks are not the final product, but merely an initial draft that can be later improved. Humans can edit the slides in order to: remove or add some sentences; add figures and tables; rectify some grammar and coherence problems in the text; etc.

5.4 Human Evaluation of Slides

The final step of this work is to present the slide decks to a group of people for human evaluation. This type of evaluation is described in section 2.4 and essentially involves showing a group of people the original text and the resulting slides, and then asking them questions about those slides. This type of evaluation seeks to evaluate every aspect of the slide decks, such as information organization, and not just the summary. Furthermore, it also allows for the summary to be evaluated based on the information that it has and not just the similarity that it has with a golden summary. However, every evaluation that is going to be received is subjective, and it can be highly variable since different people may have different opinions.

In order to create a process of human evaluation, the first step was to choose the slides to test. This is due to the fact that many experiments were conducted as part of this report, and providing all of them to people would necessitate a significant amount of effort on their part to evaluate the slides. There are ten articles from Wikipedia, each summarised in two languages: Portuguese and English, using a maximum of two abstractive methods (only one for Portuguese) in two different contexts (full text or separated by slides), and six different extractive methods, which already exclude SVR, that cannot be applied to the Wikipedia dataset because there is not enough data. Furthermore, these methods could also be tested with different ratios and with several combinations.

For the ratios, only the best performing ones in automatic evaluation were chosen. Even though having larger summaries could be something interesting to evaluate from the perspective of a person, this is not a feasible option because it would require that for each method evaluated, two different summaries were presented. As for combinations, these were also excluded. Even though they can improve certain metrics, the combination would change for every dataset, and since the results do not have a drastic improvement, sometimes they do not improve at all, it is more advantageous to test only the methods individually.

Regarding the language of the summaries, English was chosen. This was due to most of

the experiments in this report being made with that language. Furthermore, the evaluation metrics, except for ROUGE, and the transformers, were more adjusted to English than Portuguese. Furthermore, Multilingual T5 is the only transformer that is able to process Portuguese texts, and its automatic metrics are much below those of Distillbart or Pegasus CNN/DailyMail. Therefore, to have a better idea of what is possible to accomplish with abstractive summarization, English is the language that all the slides and respective Wikipedia articles are in. Besides that, only six of the ten articles—"Coimbra," "Europe," "Queen," "Carnation Revolution," "Cristiano Ronaldo," and "Star Wars"—were chosen, and only three slide decks were produced for each, with each slide deck having a different method. This articles were chosen to represent different themes: places-2, band-1, events-1, person-1 and movie-1. There are no technical articles due to their complexity. In appendix B there is an excerpt of the Pythagorean Theorem slide decks. As we can see, the mathematical formulas are difficult to understand because they are not presented in a clear manner. This is an extraction problem since it is all extracted as text, and formulas should have a different way of extraction that would allow them to be in the slides exactly as they are in the original text.

Three different types of methods were chosen. Since there are more extractive methods, it was decided that they would be the majority. This means that out of those three methods, two are extractive and one is abstractive. For the abstractive method, the choice relapsed on the Distillbart transformer since it was the one that presented the best results across all datasets.

The selection of extractive methods presented a more difficult decision, since there are different "best methods" for all the datasets. So, a decision was made to have one method that had already been used for slide generation and another that had only been used in more traditional summarization scenarios. For the first type, there are only two methods: QueSTS and SVR. Since SVR is not possible to apply to the Wikipedia dataset, QueSTS was the one chosen. This method is also one of the highest performing methods for the Wikipedia dataset, being the best for ROUGE 1 and L, excluding combinations, and being the second best in several other metrics (BLEURT, BERTSCORE, ROUGE 2, W, S, and SU), with only a small gap between its results and the best. So, this will also give an idea of how a method that is automatically evaluated as one of the best performs in human evaluation.

For the last method, TextRank was chosen. This is the method that was best scored in the PS5K dataset, and so its inclusion, along with Distillbart, which is the best performing method for SciDuet and Wikipedia English, will allow for an evaluation of the best methods across all datasets. Furthermore, TextRank presents very mid-range results for Wikipedia English, and is even one of the worst-performing according to certain metrics, such as ROUGE 1. This will allow for a comparison between two of the best methods in automatic evaluation (Distillbart and QueSTS) and one of the worst, which will give an idea of the different behaviours when humans and algorithms evaluate the slide decks.

In conclusion, the human evaluation tests were made up of six different articles, all written in English, each with three unique slide decks made using a different summarization method: TextRank, QueSTS, or Distillbart. In order to organise this information for people to evaluate, six forms were created, using Google Forms[1], each pertaining to an article and consisting of three sections, with each section having several questions regarding a particular slide deck. These questions are the same for every section/slide deck in every form.

[1]https://www.google.com/forms/about/

In order to develop the evaluation questions, ideas from some of the work described in section 2.4 were used as sources of inspiration. However, instead of, for example, asking respondents to rate items on a scale of one to five, the Likert scale [Likert, 1932] was employed. This consists of providing statements instead of questions that people have to rate, usually on the following scale:

- Strongly disagree (SA)
- Disagree (D)
- Neither agree nor disagree
- Agree (A)
- Strongly agree (SA)

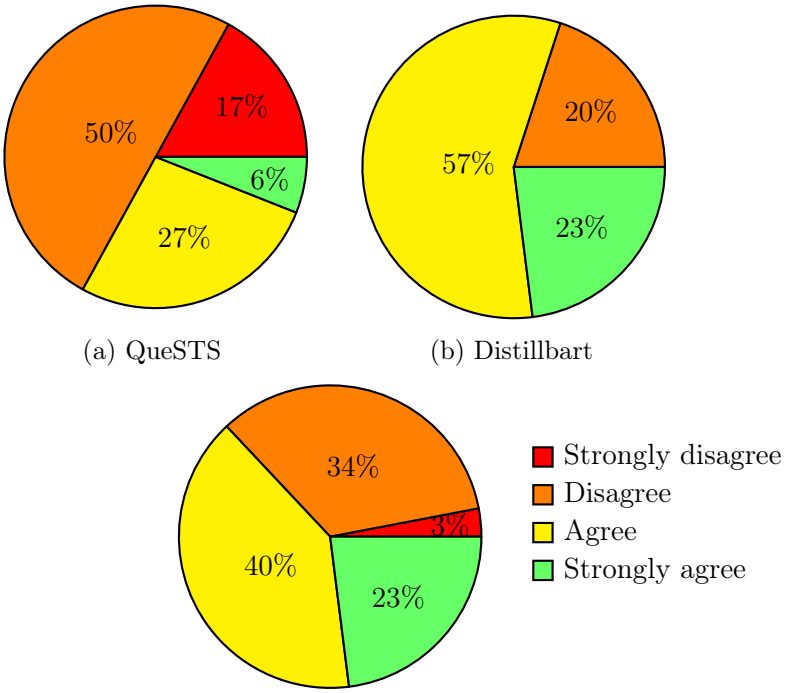
However, the third option, "Neither agree nor disagree," was removed to avoid any room for doubt in the results. In this way, the scores are always going to give positive or negative feedback.

After having the scale, the next and final step is the selection of the statements, which includes the following, inspired by Sravanthi et al. [2009]:

- I am satisfied with the amount of information in the presentation
- I am satisfied with the relevance of the information in the presentation (the selected information is the most important for the topic)
- I am satisfied with the organization of the information presented
- I am satisfied with the overall quality of the presentation

The first statement seeks to have a human perspective on the problem of the ratio that was discussed throughout this report, mainly in section 4. Even though only the methods with a lower ratio were used in the questionnaire, every slide has a different low ratio and so a different size. As for the statement regarding relevance, it seeks to know if from all the information that was presented in the original text, the most important was kept in the slide decks or not. This statement is the closest to what happens in automatic evaluation, that only compares the resulting summary to the golden one, i.e., the summary with the most relevant information. Statement three, regarding the organisation of the text, is included in order to have a better understanding if all the information is well organized, not only in terms of topics but also within the topics. The overall quality statement aims to ascertain the respondent's opinion of the presentation as a whole, sort of combining all the prior statements into one and including some additional elements that were not evaluated but that the respondent finds significant and that should be added or removed from the presentation. Finally, the last statement examines whether the slide deck achieves the primary goal of this work, which is to create slide presentations that would serve as a good foundation for the preparation of the final one.

Appendix C has an example of a form. Figures 5.1, 5.2, 5.3, 5.4 and 5.5 present the results of the forms for each statement in each method. Each statement was assessed 30 times, with the form "Coimbra" receiving eight responses, "Europe" receiving six, "Cristiano Ronaldo" receiving four, "Carnation Revolution" receiving six, and "Star Wars" and



(a) QueSTS (b) Distillbart

(c) TextRank

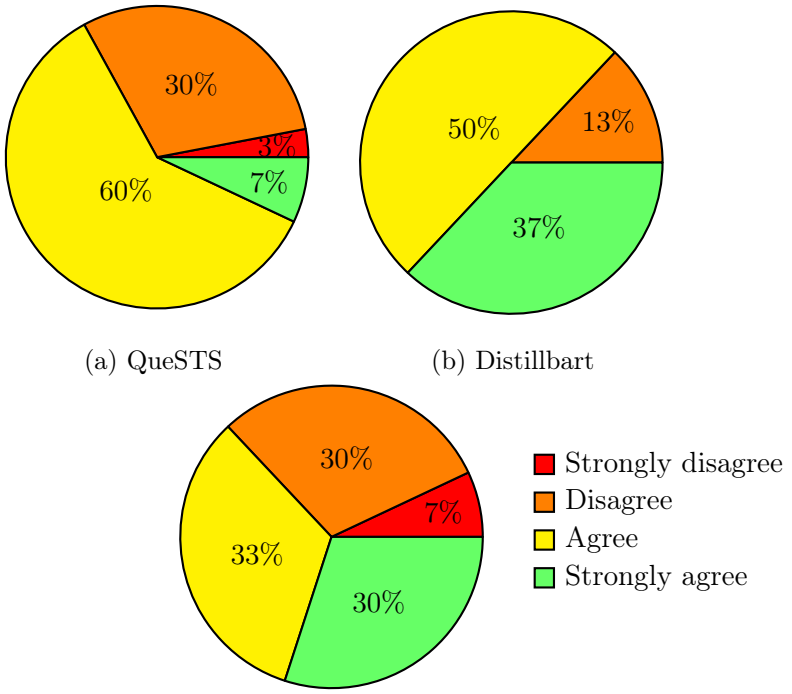
Figure 5.1: I am satisfied with the amount of information in the presentation

"Queen" receiving three each. All the responses were combined to increase the number of responses for each method and produce more reliable results.

The best performing method for the statement regarding the amount of information in the presentation, as it is possible to see in figure 5.1, is Distillbart, with 80% of positive classifications, while TextRank has 63% and QueSTS has only 33%. In terms of the quantity of existing information, Distillbart is between QueSTS and TextRank, not having as much information as TextRank but having more than QueSTS. Therefore, the summaries' quantity should be changed to one that is more in line with the Distillbart, especially for QueSTS, which has a really low score, indicating a big information gap. Furthermore, since TextRank has the same ratio as many of the other employed methods, it is not necessary to test them to know that this problem will also be verified in those methods.

As for the statement regarding the relevancy of the information in slides, as is seen in figure 5.2, the best performing method is, once again, Distillbart, with 87% of positive scores, while TextRank has 63% and QueSTS has 67%. Even though, QueSTS has more positive reviews, TextRank has more "Strongly Agree" evaluations than QueSTS. TextRank has 30% while QueSTS has only 7%.

In the first statement regarding the quantity of available information, QueSTS only obtained a positive score of 33% and now regarding the relevancy of that information, the method was able to obtain 67% of positive evaluations. This indicates that, despite the fact that QueSTS presents limited information, what is presented is important. This makes sense given the extensive information filtering required to select the few sentences that appear in the final slides. However, in terms of information relevance, QueSTS is still not the best method; Distillbart holds that distinction. This might be because Distillbart has training and so is able to pinpoint the most important information more easily. Furthermore, this is also the only asbtractive method, which means that it can cut from sentences



(a) QueSTS (b) Distillbart

(c) TextRank

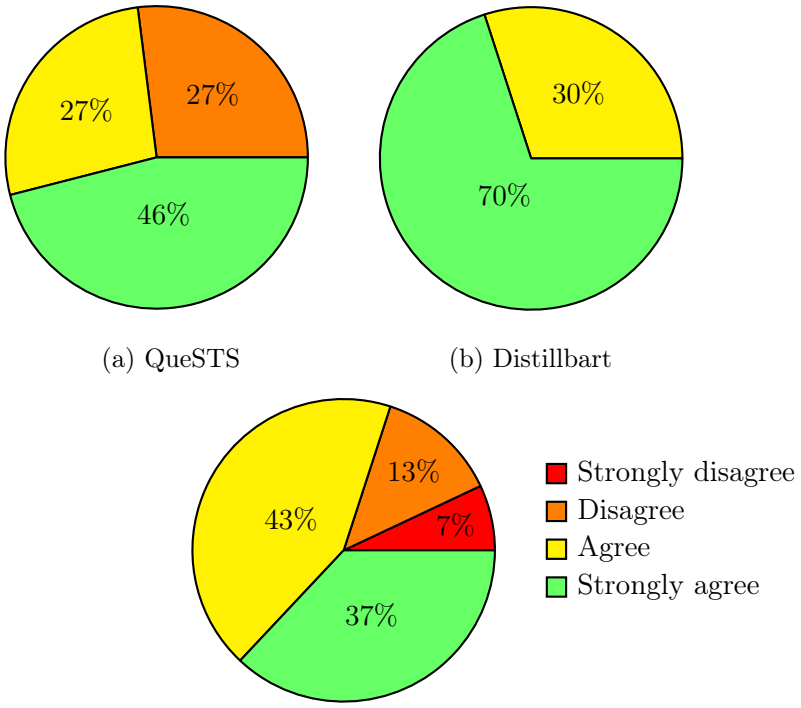
Figure 5.2: I am satisfied with the relevance of the information in the presentation (the selected information is the most important for the topic)

information that is not important. This will also decrease the total size of existent information, which results in more relevancy in fewer words. As a result, the first statement, which refers to information quantity, and the second statement, which deals with relevancy, will receive higher scores. The opposite happens for the extractive methods, which are unable to cut sentences, and therefore they might be adding a sentence that only has a part that is relevant, which will increase the quantity and decrease the relevancy.

Figure 5.3 presents the results regarding the organisation of the information. Distillbart is, once again, the best method, this time without any negative evaluations, with 70% of the responders strongly agreeing that the information is well organized. After that, the best is TextRank with 80% of positive scores (43%: "Strongly Agree" and 37%: "Agree") and then QueSTS with 73% of positive scores (46%: "Strongly Agree" and 27%: "Agree"). This is the statement with the best scores, which might be due to the information in the original articles from Wikipedia already being well organized. The way the information is presented on each slide may be the reason why the scores from the various methods differ. Distillbart presents a more coherent type of text, which may aid in the organisation of ideas in contrast to TextRank and QueSTS, which only extract full sentences from the original text without regard to context or idea order.

As for the statement regarding the overall quality of the presentation, its evaluation results can be found in figure 5.4. The best evaluated method for this statement is Distillbart, followed by TextRank, and then QueSTS, with 77%, 50% and 44% of positive ratings, respectively. While both TextRank and Distillbart receive the same proportion of "Strongly Agree" and "Strongly Disagree" ratings, their ratings differ in the other categories, with Distillbart receiving more "Agree" ratings than TextRank.

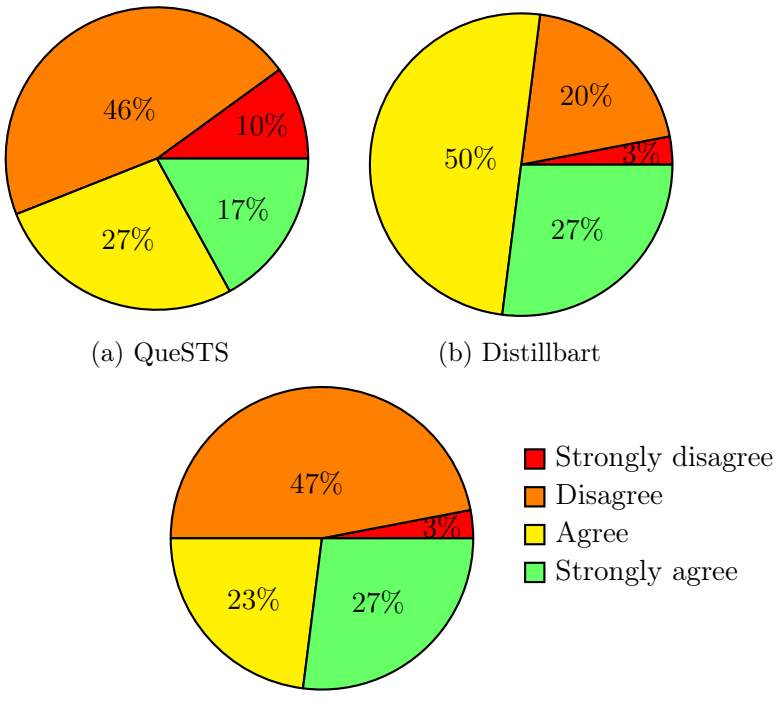
This statement takes into consideration all aspects of a slide deck, including the ones evaluated in the previous statements. So, taking into consideration the evaluations ob-



(a) QueSTS (b) Distillbart

(c) TextRank

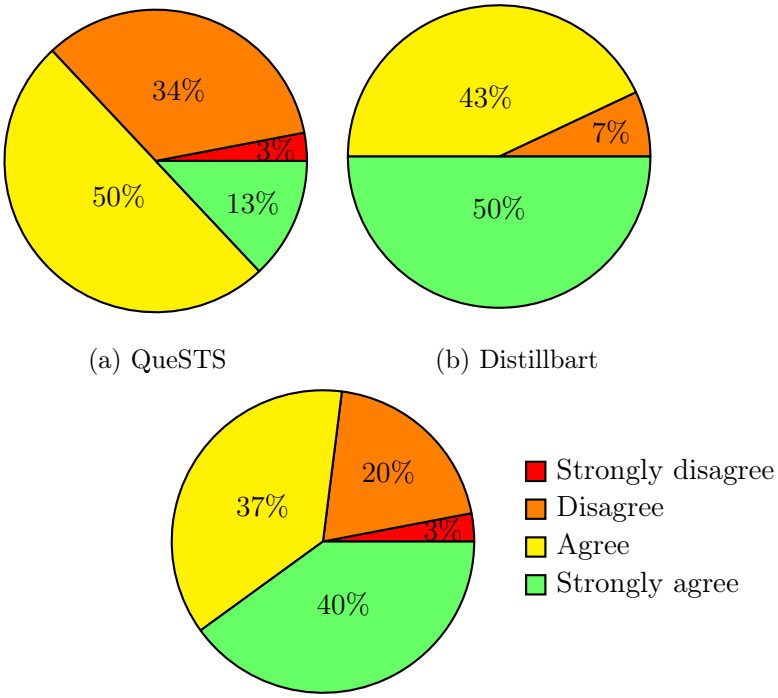
Figure 5.3: I am satisfied with the organization of the information presented



(a) QueSTS (b) Distillbart

(c) TextRank

Figure 5.4: I am satisfied with the overall quality of the presentation



(a) QueSTS (b) Distillbart

(c) TextRank

Figure 5.5: This presentation is a good starting point to prepare for the final presentation

tained for the other statements, these scores are expected, with the most preferable method for people being the abstractive method: Distillbart.

Finally, in figure 5.5 are presented the results of the evaluation of the final statement regarding the main objective of this work, i.e., producing slides that are a good starting point to prepare for the final presentation. Distillbart is, again, the best evaluated method, receiving the highest percentage of positive evaluations (93%) followed by TextRank (77%) and QueSTS (63%). Despite Distillbart being the best, the other methods still show a majority of favourable results, demonstrating that all the evaluated methods produce presentations that are a good place to start when getting ready for the final one.

In table 5.17 there is a compendium of all the human evaluation results. The order of statements is the same as the figures before. As for the answers, it is used an abbreviation, as it is on the items above.

Method	Quantity				Relevance				Organization			
	SD	D	A	SA	SD	D	A	SA	SD	D	A	SA
QueSTS	5	15	8	2	1	9	18	2	0	8	8	14
Distillbart	0	6	17	7	0	4	15	11	0	0	9	21
TextRank	1	10	12	7	2	9	10	9	2	4	13	11

Method	Quality				Starting Point			
	SD	D	A	SA	SD	D	A	SA
QueSTS	3	14	8	5	1	10	15	4
Distillbart	1	6	15	8	0	2	13	15
TextRank	1	14	7	8	1	6	11	12

Table 5.17: Compendium of all the human evaluation results.

5.5 Main Conclusions

This chapter presents an automatic and human evaluation of summaries and slides, respectively, generated through some Wikipedia articles. In automatic evaluation, the best performing method was Distillbart, followed by (in order, loosely): Lexical Chains, QueSTS, LexRank, LSA, TextRank, and TF-IDF (the last three are all very close in value, which makes it difficult to determine which is best). As for human evaluation, the best evaluated method was also Distillbart, followed by (in order): TextRank, and QueSTS. In this evaluation, only these three methods were tested. It is also important to take into consideration that a different number of articles were used in automatic and human evaluation. So, there might be some articles that might change the outcomes, for better or worse.

QueSTS outperformed TextRank in automatic evaluation, but this did not hold true for human evaluation. The issue could be caused by the small size of the golden summary, which is used to assess the generated summaries. As a result, QueSTS summary received the highest rating because it also has a small size, which fits with the golden summary. The best method, however, was based on Distillbart, which was deemed to be the best for both types of evaluation. So, even though in the state of the art, abstractive methods have very little use for slide generation, this appears to be a very interesting option to explore and apply to this problem.

Chapter 6

Conclusion

This thesis presents a study of the two steps involved in the automatic generation of slides: summarization and slide generation. Summarization is the one that receives the most attention since it is the first and most diverse step.

At the beginning of this thesis, there is a compendium of several summarization methods. Some are automatically tested in three English datasets: two for slide generation and one for summarization. In those tests, extractive and abstractive summarization methods were chosen in order to understand which was the best-suited approach. However, each dataset had a different best method. Two of the three datasets included text and the corresponding slide decks, while the third included text and the corresponding summary. Only one of those datasets—with slides as the goal—the best performance was achieved with an extractive method, with the others producing better results when abstractive methods were used. Even within extractive methods, the best performing method varies according to the data.

In addition to automatic evaluation, human evaluation was also carried out. The people that evaluated the slides preferred the abstractive methods to the extractive methods. This proves that these methods are very interesting and warrant a larger research than what has been done so far since they are rarely used for slide generation. However, these methods have a limitation when compared to the extractive ones, since the majority of extractive methods, in contrast to the abstractive methods, are unsupervised, which means they do not require any training. As a result, they can summarise any text, regardless of the topic or language. This was demonstrated by some tests using a Portuguese dataset. A very limited multilingual transformer and some unsupervised extractive methods were tested on this dataset, and the results of the extractive methods were noticeably better.

As such this thesis offered a number of contributions to the automatic slide generation problem. Several methods, including extractive and abstractive, as well as supervised and unsupervised ones, were tested in order to provide a comparison between them. In an effort to improve the performance of the methods, a number of extractive methods were combined. Unsupervised extractive methods and abstractive methods were tested in Portuguese in addition to English, which was tested in every method. Three datasets were used to test the methods; one of them was made especially for this study and consisted of summaries rather than slides, even if slides were then generated. Furthermore, three automatic evaluation metrics were used, and a human evaluation was conducted for the final slide decks. Additionally, a scientific paper [Costa et al., 2022], on the experiments with the unsupervised methods, was written and accepted for publication.

Despite all the possible improvements and directions left to explore, the balance of this work is positive. A range of methods is now easily available for the automatic generation of slides, with several conclusions taken on their advantages and limitations. This will definitely contribute to accelerate the process of slide production, which is currently a completely manual process.

In order to improve this process, several directions for future work were identified. This includes options for unsupervised abstractive summarization that may be explored along with other interesting methods that were not applied in this work. Examples include QA methods (described in 3.2), neural network methods (described in 3.1.5), such as RNNs, and discourse-based methods (described in 3.1.2), among other not covered in this thesis. Additionally, summarization can be expanded to include more than one document, and a method for handling texts with more complex language, such as mathematical formulas, can be studied. As for slide decks, future work might entail adding visual aids like tables and images as well as coming up with alternative ways to arrange the sentences. A method that can be used to accomplish this is topic modelling [Kherwa and Bansal, 2020]. This unsupervised machine learning method scans one or more documents, looks for patterns in the sentences, and groups the various words into topics. This would enable us to organise the data without relying on section headings in a document.

References

Belz, A. and Kow, E. (2010). Comparing rating scales and preference judgements in language evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics.

Bhandare, A. A., Awati, C. J., and Kharade, S. (2016). Automatic era: Presentation slides from academic paper. *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, pages 809–814.

Bird, S., Dale, R., Dorr, B., Gibson, B. R., Joseph, M. T., Kan, M.-Y., Lee, D., Powley, B., Radev, D. R., and Tan, Y. F. (2008). The acl anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *LREC*.

Chen, D. and Manning, C. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation.

Christian, H., Agus, M., and Suhartono, D. (2016a). Single document automatic text summarization using term frequency-inverse document frequency (tf-idf). *ComTech: Computer, Mathematics and Engineering Applications*, 7:285.

Christian, H., Agus, M. P., and Suhartono, D. (2016b). Single document automatic text summarization using term frequency-inverse document frequency (tf-idf). *ComTech*, 7(4).

Chung, H. W., Févry, T., Tsai, H., Johnson, M., and Ruder, S. (2020). Rethinking embedding coupling in pre-trained language models. *CoRR*, abs/2010.12821.

Collins, A. and Loftus, E. (1975). A spreading activation theory of semantic processing. *Psychological Review*, 82:407–428.

Cornegruta, S., Bakewell, R., Withey, S. J., and Montana, G. (2016). Modelling radiological language with bidirectional long short-term memory networks. In *Louhi@EMNLP*.

Costa, M. J., Amaro, H., and Gonçalo Oliveira, H. (2022). Unsupervised summarization approaches for slide generation. In *Proceedings of XXIV International Symposium on Computers in Education (SIIE)*, page (accepted for publication).

Councill, I., Giles, C. L., and Kan, M.-Y. (2008). ParsCit: an open-source CRF reference string parsing package. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Dean, M., Schreiber, A. T., Bechofer, S. K., van Harmelen, F., Hendler, J. A., Horrocks, I., MacGuinness, D., Patel-Schneider, P. F., and Stein, L. A. (2004). Owl web ontology language - reference.

Deerwester, S. C., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the Association for Information Science and Technology*, 41:391–407.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. ACL.

Dinan, E., Logacheva, V., Malykh, V., Miller, A., Shuster, K., Urbanek, J., Kiela, D., Szlam, A., Serban, I., and et al., R. L. (2019). The second conversational intelligence challenge (convai2).

Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Ermakova, L., Cossu, J. V., and Mothe, J. (2019). A survey on evaluation of summarization methods. *Information Processing Management*, 56(5):1794–1814.

Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2021). SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., and Auli., M. (2019). Eli5: Long form question answering.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.

Fu, T.-J., Wang, W. Y., McDuff, D. J., and Song, Y. (2021). Doc2ppt: Automatic presentation slides generation from scientific documents. *ArXiv*, abs/2101.11796.

Gokaslan, A. and Cohen., V. (2019). Openwebtext corpus.

Gupta, S. and Gupta, S. K. (2019). Abstractive summarization: An overview of the state of the art. *Expert Syst. Appl.*, 121:49–65.

Hanaue, K., Ishiguro, Y., and Watanabe, T. (2012). Composition method of presentation slides using diagrammatic representation of discourse structure. *Int. J. Knowl. Web Intell.*, 3:237–255.

Hasan, T., Bhattacharjee, A., Islam, M. S., Mubasshir, K., Li, Y.-F., Kang, Y.-B., Rahman, M. S., and Shahriyar, R. (2021). XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Hashemi, M., Azizinezhad, M., and Farokhi, M. (2012). Power point as an innovative tool for teaching and learning in modern classes. *Procedia - Social and Behavioral Sciences*, 31:559–563. World Conference on Learning, Teaching Administration - 2011.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Hu, Y. and Wan, X. (2013). Ppsgen: Learning to generate presentation slides for academic papers. In *IJCAI*.

Iskender, N., Polzehl, T., and Möller, S. (2021). Reliability of human evaluation for text summarization: Lessons learned and challenges ahead. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 86–96, Online. Association for Computational Linguistics.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.

Kherwa, P. and Bansal, P. (2020). Topic modeling: a comprehensive review. *EAI Endorsed transactions on scalable information systems*, 7(24).

Kostadinov, S. (2019). Understanding gru networks.

Koupaee, M. and Wang, W. Y. (2018). Wikihow: A large scale text summarization dataset. *CoRR*, abs/1810.09305.

KUROHASHI, S. (1994). Improvements of japanese morphological analyzer juman. *Proceedings of The International Workshop on Sharable Natural Language, 1994*, pages 22–28.

Kurohashi, S. and Nagao, M. (1994). A syntactic analysis method of long japanese sentences based on the detection of conjunctive structures. *Comput. Linguistics*, 20:507–534.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Li, D.-W., Huang, D., Ma, T., and Lin, C.-Y. (2021). Towards topic-aware slide generation for academic papers with unsupervised mutual learning. In *AAAI*.

Likert, R. (1985 - 1932). *A technique for the measurement of attitudes / by Rensis Likert.* Archives of psychology ; no. 140. [s.n.], New York.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM J. Res. Dev.*, 2:159–165.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *ACL*.

Mathivanan, H., Jayaprakasam, M., Prasad, K. G., and Geetha, T. V. (2009). Document summarization and information extraction for generation of presentation slides. *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, pages 126–128.

McGuinness, D. L. (2004). Owl web ontology language overview.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into text. In *Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. ACL.

Morita, T., Fukuta, N., Izumi, N., and Yamaguchi, T. (2006). Doddle-owl: A domain ontology construction tool with owl. volume 4185, pages 537–551.

Nagao, K. and Hasida, K. (1998). Automatic text summarization based on the global document annotation. In *COLING-ACL*.

Nagel, S. (2016). Cc-news.

Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.

Narayan, S., Cohen, S. B., and Lapata, M. (2018). Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Nenkova, A. and McKeown, K. (2012). A Survey on Text Summarization Techniques. In Aggarwal, C. C. and Zhai, C., editors, *Mining Text Data*, pages 43–76. Springer US, Boston, Massachusetts, USA.

Pavlichenko, N., Stelmakh, I., and Ustalov, D. (2021). Vox populi, vox diy: Benchmark dataset for crowdsourced audio transcription.

Phi, M. (2020). Illustrated guide to lstm's and gru's: A step by step explanation.

Pletenev, S. (2021). Noisy text sequences aggregation as a summarization subtask. In *Proceedings of the 2nd Crowd Science Workshop: Trust, Ethics, and Excellence in Crowdsourced Data Management at Scale*, pages 15–20, Copenhagen, Denmark.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer.

Rajpurkar, P., Zhang, J., Lopyrev, K., , and Liang., P. (2016). Squad: 100,000+ questions for machine comprehension of text.

Reed, S. (2008). Spreading activation.

Rothe, S., Narayan, S., and Severyn, A. (2019). Leveraging pre-trained checkpoints for sequence generation tasks. *CoRR*, abs/1907.12461.

Sariki, T., Sariki, Kumar, B., and Ragala, R. (2014). Effective classroom presentation generation using text summarization. *IJCTA*, 5.

Sathiyamurthy, K. and Geetha, T. V. (2012). Automatic organization and generation of presentation slides for e-learning. *Int. J. Distance Educ. Technol.*, 10:35–52.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Sefid, A., Mitra, P., Wu, J., and Giles, C. L. (2021a). Extractive research slide generation using windowed labeling ranking. *ArXiv*, abs/2106.03246.

Sefid, A., Wu, J., Mitra, P., and Giles, C. L. (2019). Automatic slide generation for scientific papers. In *SciKnow@K-CAP*.

Sefid, A., Wu, J., Mitra, P., and Giles, C. L. (2021b). Extractive research slide generation using windowed labeling ranking. *CoRR*, abs/2106.03246.

Sellam, T., Das, D., and Parikh, A. P. (2020). BLEURT: learning robust metrics for text generation. *CoRR*, abs/2004.04696.

Sethi, P., Sonawane, S. S., Khanwalker, S., and Keskar, R. B. (2017). Automatic text summarization of news articles. *2017 International Conference on Big Data, IoT and Data Science (BID)*, pages 23–29.

Shaikh, P. J. and Deshmukh, R. A. (2016). Automatic slide generation for academic paper using ppsgen method. *International Journal of Technical Research and Applications*, pages 199–203.

Shibata, T. and Kurohashi, S. (2005). Automatic slide generation based on discourse structure analysis. In *IJCNLP*.

Shleifer, S. and Rush, A. M. (2020). Pre-trained summarization distillation. *CoRR*, abs/2010.13002.

Sravanthi, M., Chowdary, C. R., and Kumar, P. S. (2008). Quests: A query specific text summarization system. In *FLAIRS Conference*.

Sravanthi, M., Chowdary, C. R., and Kumar, P. S. (2009). Slidesgen: Automatic generation of presentation slides for a technical paper using summarization. In *FLAIRS Conference*.

Steinberger, J. and Jezek, K. (2004). Using latent semantic analysis in text summarization and summary evaluation. volume 4.

Steinberger, J. and Jezek, K. (2009). Evaluation measures for text summarization. *Computing and Informatics*, 28:251–275.

Sun, E., Hou, Y., Wang, D., Zhang, Y., and Wang, N. X. (2021a). D2s: Document-to-slide generation via query-based text summarization. In *NAACL*.

Sun, E., Hou, Y., Wang, D., Zhang, Y., and Wang, N. X. R. (2021b). D2S: document-to-slide generation via query-based text summarization.

Swapna. (2022). Convolutional neural network: Deep learning.

Syamili, S. and Abraham, A. (2017). Presentation slides generation from scientific papers using support vector regression. *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 286–291.

Utiyama, M. and Hasida, K. (1999). Automatic slide presentation from semantically annotated documents. In *COREF@ACL*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.

Wang, S., Wan, X., and Du, S. (2017). Phrase-based presentation slides generation for academic papers. In *AAAI*.

Williams, A., Nangia, N., and Bowman., S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference.

Zhang, J., Zhao, Y., Saleh, M., and Liu, P. J. (2019). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020). Bertscore: Evaluating text generation with bert.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724.

Appendices

Appendix A: Slide Decks Examples

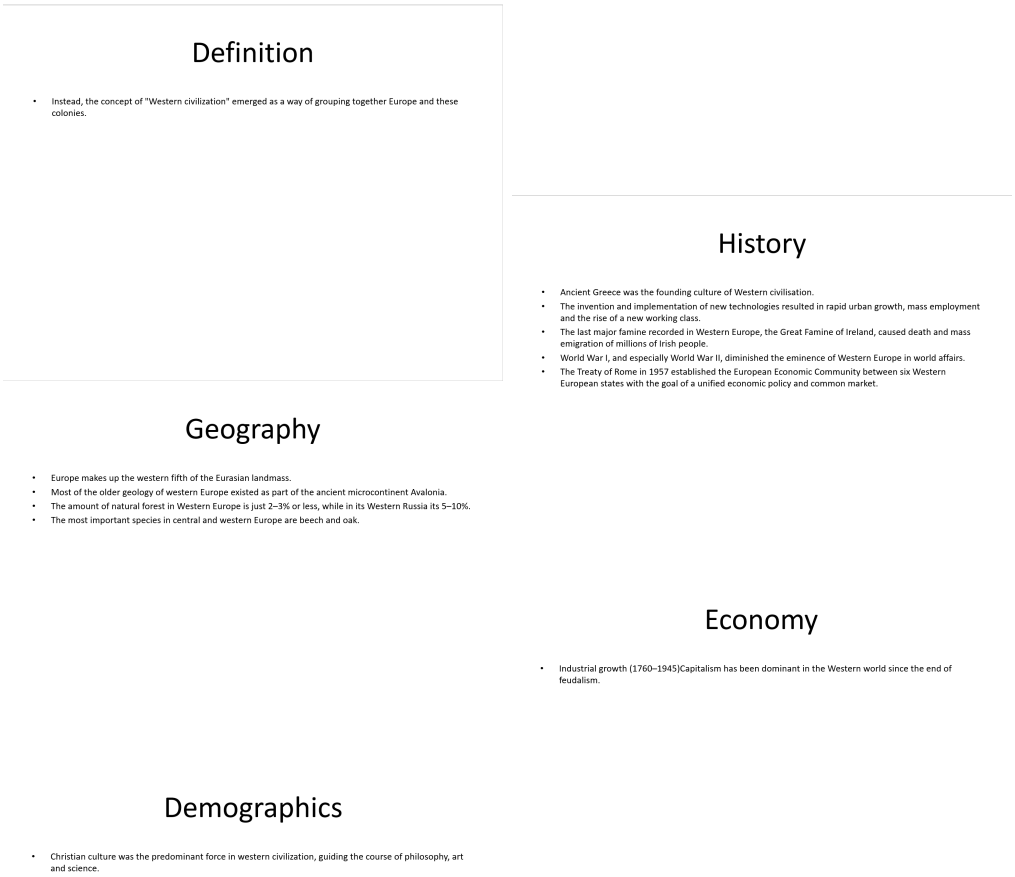


Figure 1: Every slide generated with QueSTS for the article “Europe”, in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Europe

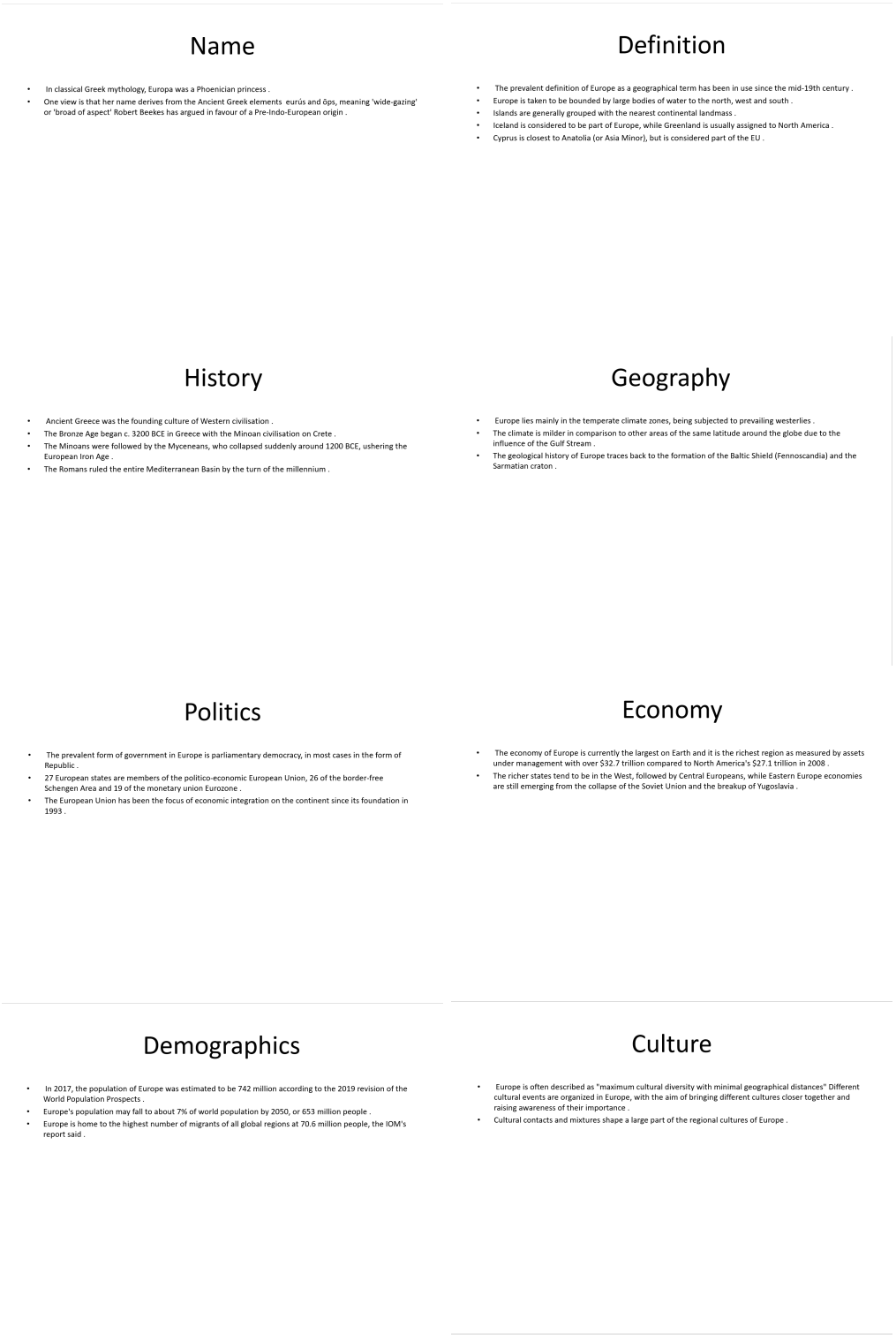


Figure 2: Slides generated with Distillbart for the article "Europe", in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Europe

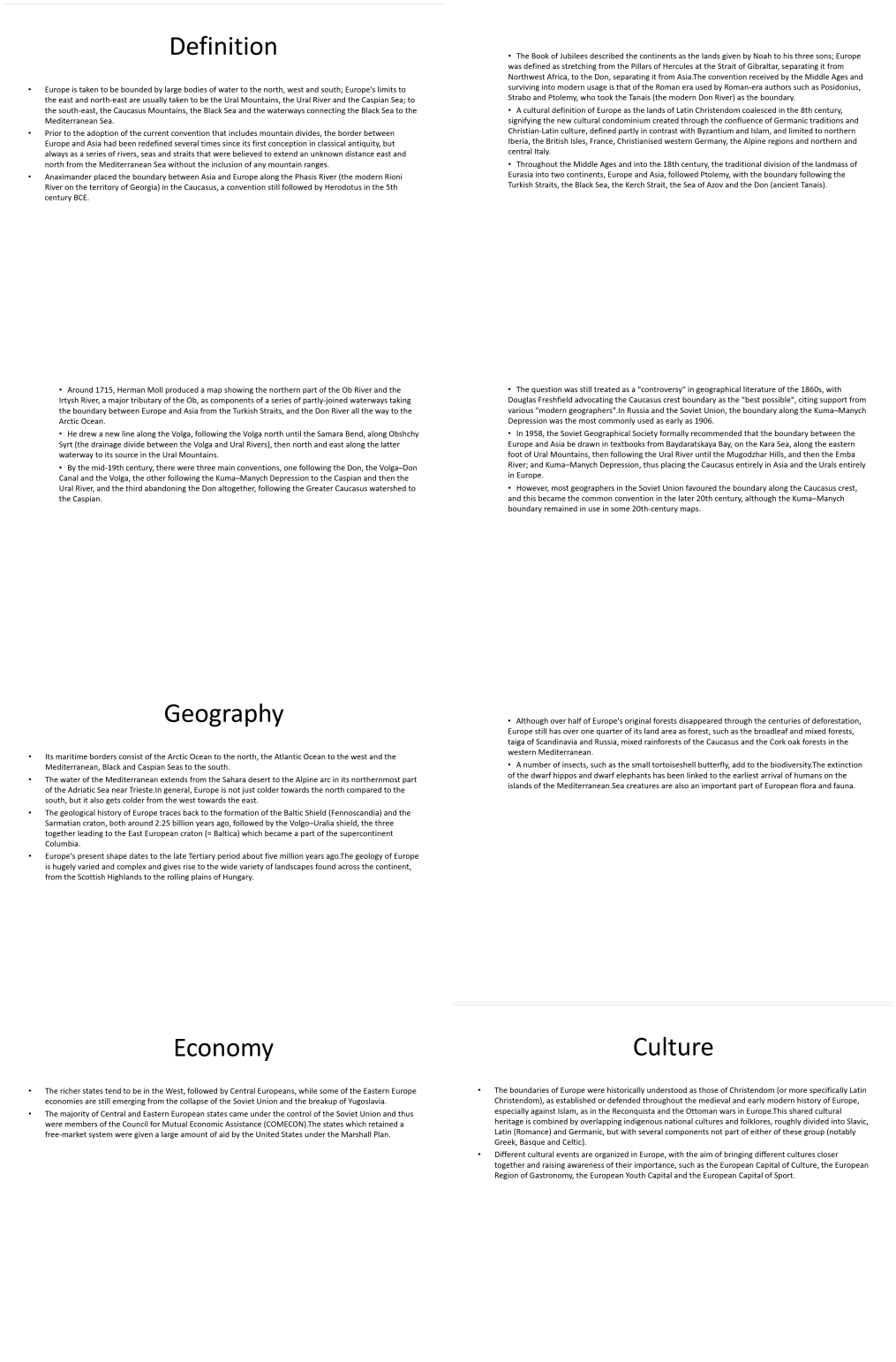


Figure 3: Slides generated with TextRank for the article “Europe”, in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Europe

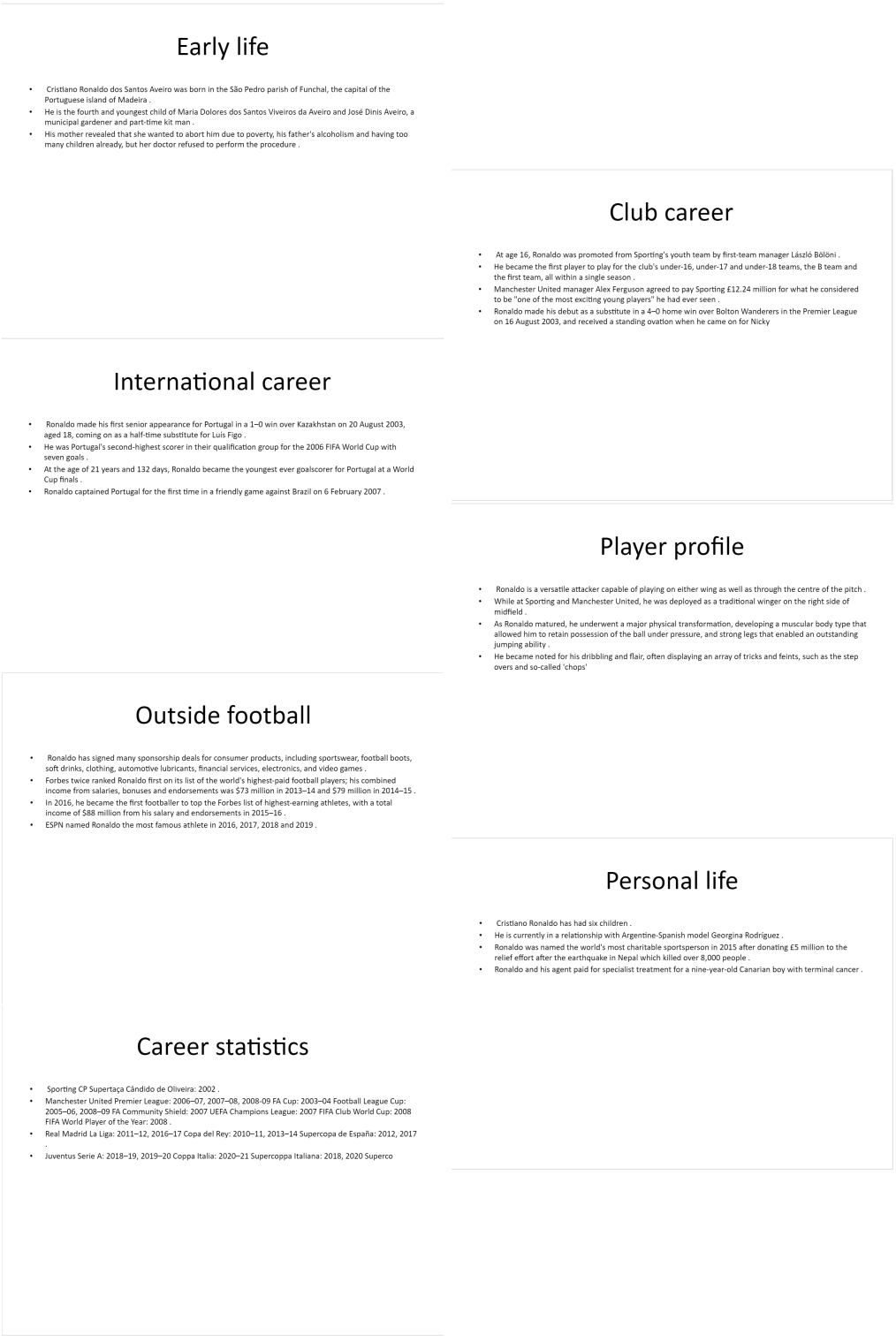


Figure 4: Every slide generated with Distillbart for the article "Cristiano Ronaldo", in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Cristiano$_R$onaldo

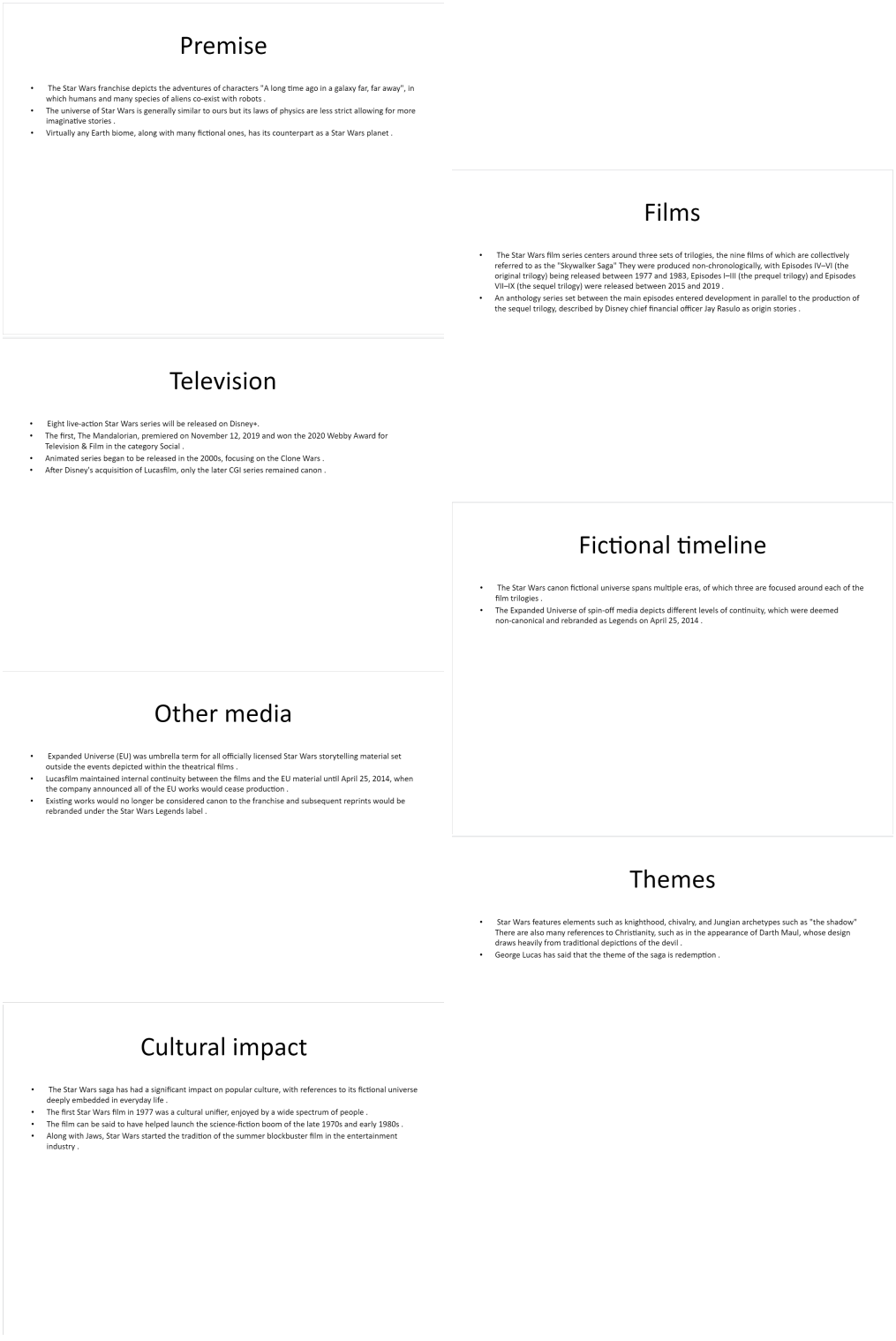


Figure 5: Every slide generated with Distillbart for the article "Star Wars", in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Star$_W ars$

Figure 6: Every Slide generated with TextRank for the article “Coimbra”, in the Portuguese Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://pt.wikipedia.org/wiki/Coimbra

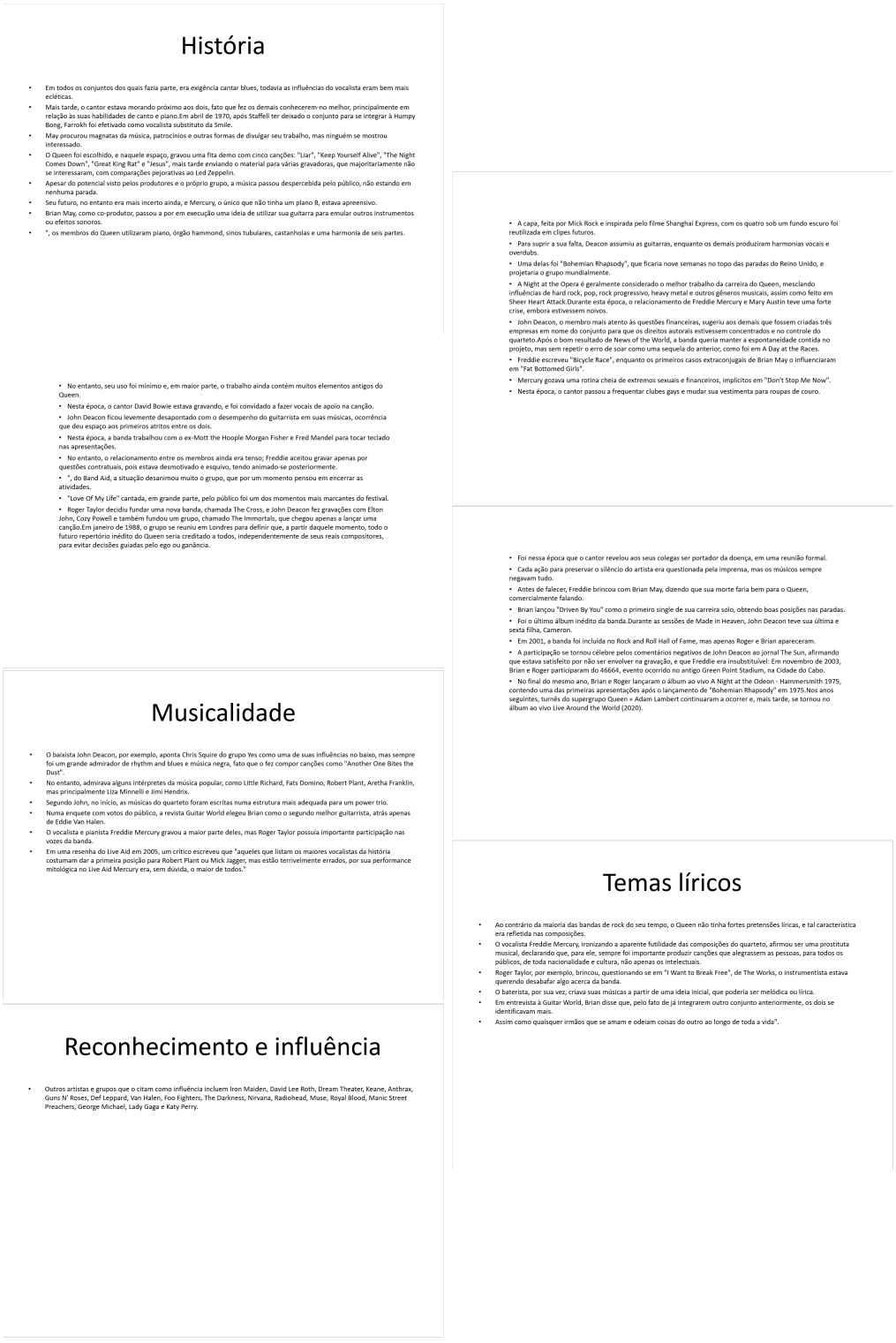


Figure 7: Every Slide generated with TextRank for the article "Queen", in the Portuguese Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://pt.wikipedia.org/wiki/Queen

Appendix B: Pythagorean Theorem Slide Decks

Other forms of the theorem

- If c denotes the length of the hypotenuse and a and b denote the two lengths of the legs of a right triangle, then the Pythagorean theorem can be expressed as the Pythagorean equation: a 2 + b 2 = c 2 .
- If only the lengths of the legs of the right triangle are known but not the hypotenuse, then the length of the hypotenuse can be calculated with the equation c = a 2 + b 2 .
- A generalization of this theorem is the law of cosines, which allows the computation of the length of any side of any triangle, given the lengths of the other two sides and the angle between them.

Consequences and uses of the theorem

- The Pythagorean theorem has, a 2 + b 2 = c 2 {\displaystyle a^{2}+b^{2}=c^{2}} while the reciprocal Pythagorean theorem or the upside down Pythagorean theorem relates the two legs a , b {\displaystyle a,b} to the altitude d {\displaystyle d} , 1 a 2 + 1 b 2 = 1 d 2 {\displaystyle {\frac {1}{a^{2}}}+{\frac {1}{b^{2}}}={\frac {1}{d^{2}}}} The equation can be transformed to, 1 (x z) 2 + 1 (y z) 2 = 1 (x y) 2 {\displaystyle {\frac {1}{(xz)^{2}}}+{\frac {1}{(yz)^{2}}}={\frac {1}{(xy)^{2}}}} where x 2 + y 2 = z 2 {\displaystyle x^{2}+y^{2}=z^{2}} for any non-zero real x , y , z {\displaystyle x,y,z} .
- More generally, in Euclidean n-space, the Euclidean distance between two points, A = (a 1 , a 2 , … , a n) {\displaystyle A\=\(a_{1},a_{2},\dots ,a_{n}\)} and B = (b 1 , b 2 , … , b n) {\displaystyle B\=\(b_{1},b_{2},\dots ,b_{n}\)} , is defined, by generalization of the Pythagorean theorem, as: (a 1 − b 1) 2 + (a 2 − b 2) 2 + ⋯ + (a n − b n) 2 = ∑ i = 1 n (a i − b i) 2 .
- If instead of Euclidean distance, the square of this value (the squared Euclidean distance, or SED) is used, the resulting equation avoids square roots and is simply a sum of the SED of the coordinates: (a 1 − b 1) 2 + (a 2 − b 2) 2 + ⋯ + (a n − b n) 2 = ∑ i = 1 n (a i − b i) 2 .

Proofs using constructed squares

- In the first square, the triangles are placed such that the corners of the square correspond to the corners of the right angle in the triangles, forming a square in the center whose sides are length c. Each square has an area of both (a + b) 2 {\displaystyle (a+b)^{2}} and 2 a b + c 2 {\displaystyle 2ab+c^{2}} , with 2 a b {\displaystyle 2ab} representing the area of the four triangles.
- The four triangles and the square side c must have the same area as the larger square, (b + a) 2 = c 2 + 4 a b 2 = c 2 + 2 a b , {\displaystyle (b+a)^{2}=c^{2}+4{\frac {ab}{2}}=c^{2}+2ab,} giving c 2 = (b + a) 2 − 2 a b = b 2 + 2 a b + a 2 − 2 a b = a 2 + b 2 .

Generalizations

- This was known by Hippocrates of Chios in the 5th century BC, and was included by Euclid in his Elements: If one erects similar figures (see Euclidean geometry) with corresponding sides on the sides of a right triangle, then the sum of the areas of the ones on the two smaller sides equals the area of the one on the larger side.
- While Euclid's proof only applied to convex polygons, the theorem also applies to concave polygons and even to similar figures that have curved boundaries (but still with part of a figure's boundary being the side of the original triangle).The basic idea behind this generalization is that the area of a plane figure is proportional to the square of any linear dimension, and in particular is proportional to the square of the length of any side.
- (See also Einstein's proof by dissection without rearrangement) The Pythagorean theorem is a special case of the more general theorem relating the lengths of sides in any triangle, the law of cosines: a 2 + b 2 − 2 a b cos ⁡ θ = c 2 , {\displaystyle a^{2}+b^{2}-2ab\cos {\theta }=c^{2},} where θ {\displaystyle \theta } is the angle between sides a {\displaystyle a} and b {\displaystyle b} .

- The upper figure shows that for a scalene triangle, the area of the parallelogram on the longest side is the sum of the areas of the parallelograms on the other two sides, provided the parallelogram on the long side is constructed as indicated (the dimensions labeled with arrows are the same, and determine the sides of the bottom parallelogram).
- The left green parallelogram has the same area as the left, blue portion of the bottom parallelogram because both have the same base b and height h. However, the left green parallelogram also has the same area as the left green parallelogram of the upper figure, because they have the same base (the upper left side of the triangle) and the same height normal to that side of the triangle.
- A substantial generalization of the Pythagorean theorem to three dimensions is de Gua's theorem, named for Jean Paul de Gua de Malves: If a tetrahedron has a right angle corner (like a corner of a cube), then the square of the area of the face opposite the right angle corner is the sum of the squares of the areas of the other three faces.
- Then the square of the volume of the hypotenuse of S is the sum of the squares of the volumes of the n legs.

- The "hypotenuse" is the base of the tetrahedron at the back of the figure, and the "legs" are the three sides emanating from the vertex in the foreground.
- In a different wording: Given an n-rectangular n-dimensional simplex, the square of the (n − 1)-content of the facet opposing the right vertex will equal the sum of the squares of the (n − 1)-contents of the remaining facets.
- Here the vectors v and w are akin to the sides of a right triangle with hypotenuse given by the vector sum v + w. This form of the Pythagorean theorem is a consequence of the properties of the inner product: ‖ v + w ‖ 2 = ⟨ v + w , v + w ⟩ = ⟨ v , v ⟩ + ⟨ w , w ⟩ + ⟨ w , v ⟩ = ‖ v ‖ 2 + ‖ w ‖ 2 , {\displaystyle \left\|\mathbf {v} +\mathbf {w} \right\|^2=\langle \mathbf {v+w} ,\ \mathbf {v+w} \rangle =\langle \mathbf {v} ,\ \mathbf {v} \rangle +\langle \mathbf {w} ,\ \mathbf {w} \rangle +\langle \mathbf {v} ,\ \mathbf {w} \rangle +\langle \mathbf {w} ,\ \mathbf {v} \rangle \ =\left\|\mathbf {v} \right\|^{2}+\left\|\mathbf {w} \right\|^{2},} where the inner products of the cross terms are zero, because of orthogonality.

- A further generalization of the Pythagorean theorem in an inner product space to non-orthogonal vectors is the parallelogram law : 2 ‖ v ‖ 2 + 2 ‖ w ‖ 2 = ‖ v + w ‖ 2 + ‖ v − w ‖ 2 , {\displaystyle 2\|\mathbf {v} \|^{2}+2\|\mathbf {w} \|^{2}=\|\mathbf {v+w} \|^{2}+\|\mathbf {v-w} \|^{2}\,,} which says that twice the sum of the squares of the lengths of the sides of a parallelogram is the sum of the squares of the lengths of the diagonals.
- Specifically, the square of the measure of an m-dimensional set of objects in one or more parallel m-dimensional flats in n-dimensional Euclidean space is equal to the sum of the squares of the measures of the orthogonal projections of the object(s) onto all m-dimensional coordinate subspaces.In mathematical terms: μ m s 2 = ∑ i = 1 x μ 2 m p i {\displaystyle \mu _{ms}^{2}=\sum _{i=1}^{x}\mathbf {\mu ^{2}} _{mp_{i}}} where: μ m {\displaystyle \mu _{m}} is a measure in m-dimensions (a length in one dimension, an area in two dimensions, a volume in three dimensions, etc.).

- However, the Pythagorean theorem remains true in hyperbolic geometry and elliptic geometry if the condition that the triangle be right is replaced with the condition that two of the angles sum to the third, say A+B = C. The sides are then related as follows: the sum of the areas of the circles with diameters a and b equals the area of the circle with diameter c. For any right triangle on a sphere of radius R (for example, if γ in the figure is a right angle), with sides a, b, c, the relation between the sides takes the form: cos ⁡ (c R) = cos ⁡ (a R) cos ⁡ (b R) .
- By expressing the Maclaurin series for the cosine function as an asymptotic expansion with the remainder term in big O notation, cos ⁡ x = 1 − x 2 2 + O (x 4) as x → 0 , {\displaystyle \cos x=1-{\frac {x^{2}}{2}}+O(x^{4})\ {\text{ as }}x\to 0\,,} it can be shown that as the radius R approaches infinity and the arguments a/R, b/R, and c/R tend to zero, the spherical relation between the sides of a right triangle approaches the Euclidean form of the Pythagorean theorem.

Figure 8: Slides generated with TextRank for the article "Pythagorean Theorem", in the English Wikipedia, as of June, 19, 2022. Read from left to right. Article Link: https://en.wikipedia.org/wiki/Pythagorean$_theorem$

Appendix C: Form Example

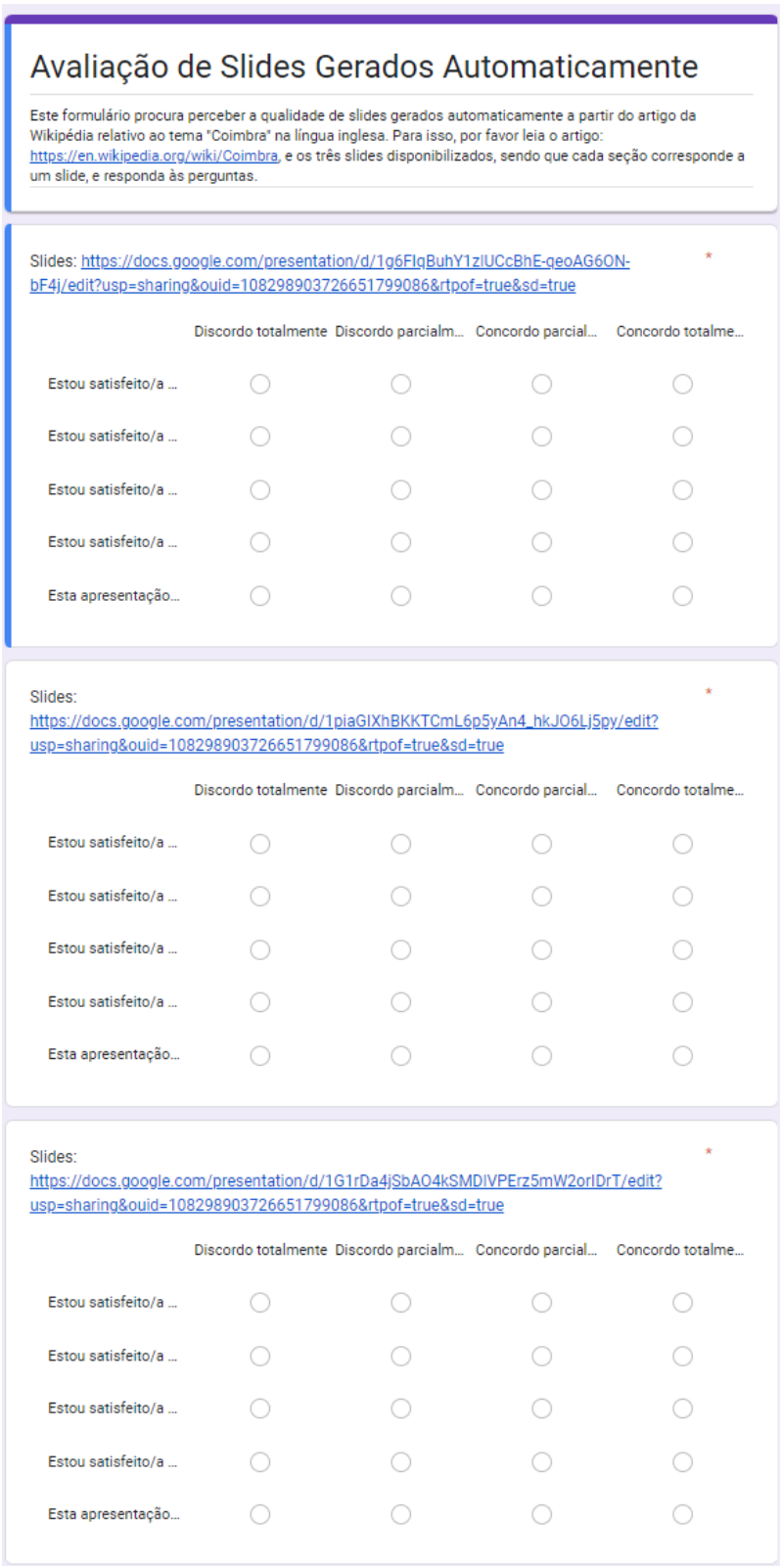


Figure 9: Form relative to English Wikipedia article "Coimbra"