People's Democratic Republic of Algeria

Ministry of Higher Education for Scientific Research

University 8 May 45 –Guelma-

Faculty of Mathematics, Computer Science and Sciences of Matter

Department of Computer Science

Master Thesis

Specialty : Computer science

Option:

Science and Technology of Information and Communication

Theme

A Text Summarization System for Faster Data Access

Presented by: Chelamet Ghada

Jury Members

Chairman: Dr.Boughareb Djalila

Supervisor: Dr. Bouressace Hassina

Examiner: Dr.Zedadra Amina

June 2023

Abstract

The increasing volume of textual information generated across various fields and domains poses a significant challenge in effectively handling and extracting valuable insights from this vast amount of data. Manual processing and analysis of every document can be time-consuming . As a result, there is a growing need for automated systems that can provide concise summaries of text documents, enabling users to quickly grasp the key points without having to go through the entire content.

To address this challenge, we have proposed a text summarization system. The goal of this system is to automatically generate condensed summaries from lengthy text documents, thereby saving time and effort in information processing. The generated summaries should capture the essence of the original text, providing a concise representation of the main ideas and important details. To achieve effective text summarization, the system utilizes a combination of approaches. One of the key approaches employed is TF-IDF combined with cosine similarity. We demonstrate the efficacy of our proposed method in summarizing articles by achieving an impressive 70 % performance in terms of ROUGE scores. This accomplishment is based on the evaluation of a substantial dataset consisting of 30 articles, each containing multiple pages of content.

Keywords : Extractive summarization, NLP,Sentence scoring, Summarization algorithms, Evaluation metrics.

Remerciements

Firstly, I would like to express my deep gratitude to God for the blessings and opportunities that have brought me to this moment. Without His guidance and grace, I would not have been able to complete this dissertation.

I would like to acknowledge and give my warmest thanks to my supervisor bouressace hassina who made this work possible. her guidance and advice carried me through all the stages of writing my project. I would also like to thank my committee members for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

I would also like to give special thanks to the best brother ever and all my family for their continuous support and understanding when undertaking my research and writing my project. Your prayer for me was what sustained me this far.

I would also like to express my gratitude to our department chair, Dr. Zineddine Kouahla, for his exceptional guidance and support throughout my academic journey. His leadership has been instrumental in shaping my experience in the Department of Computer Science, as well as the entire family of the department. My time spent there has been a beautiful experience, as I have passed all these years within its embrace. Thank you once again for everything.

Acknowledgement

This thesis is dedicated to two incredible women who have profoundly impacted my life. To my mother, who gave me life, though our time together was far too brief. I have always felt her presence in my life and her love in my heart, and I know she would have been proud of my achievements.

My grandmother sultana was my rock, she was always in my side to offer a listening ear, a warm embrace, and a word of wisdom. Her unwavering faith in me gave me the strength to overcome any obstacle and achieve my goals. Although they are no longer with me, their memory lives on in my heart and in the pages of this thesis. I hope that this work will honor their legacy and serve as a testament to their love and support.

To the man who has shaped my existence, it is because of him that I am here today, with the opportunity to pursue my passions and make a difference in the world. I am forever grateful for allah who has given me this special gift and for the endless sacrifices he has made to ensure my happiness and success. thank you father .

To my second mother nassima bouchelaghem , who took care of me for years and sacrificed so much to ensure my well-being and success, I am forever grateful for the sacrifices you made and the endless support you have provided.

I dedicate this thesis to the two amazing brothers haider and nedji and also to nidal the person who has been my constant source of guidance, support, and inspiration.I am grateful for the role he has played in my life and for the love and care he has shown me. This thesis is a tribute to all my brothers, for all the times he has believed in me and encouraged me to pursue my dreams.

To my best sister and also my best friend imene mihoub who has been my guiding light, my source of strength, and my inspiration. Her belief in me and her encouragement have given me the courage to pursue my passions and overcome any obstacle, thank you imane for being by my side all time. This thesis is dedicated to you, for all the times you have believed in me and encouraged me to chase my dreams.

I would like to express my heartfelt gratitude to the families of Chelamet and Aouamri for their love, support, and encouragement. This thesis is dedicated to all of you, for believing in me and inspiring me to reach for the stars.

Table of Contents

General Introduction 1

1 Automatic Text Summarization **3**
1 Introduction 3
2 Natural Language Processing 3
3 Objectives of NLP 3
4 Tools of NLP 4
4.1 Linguistic tools 4
4.2 Formal tools 4
4.3 Computational tools 4
5 Application of NLP 4
5.1 Machine Translation 5
5.2 Text Categorization 5
5.3 Summarization 5
5.4 Information Extraction 5
5.5 Corrector 5
6 Levels of NLP 5
6.1 Phonology 6
6.2 Morphology 7
6.3 Lexical 7
6.4 Syntactic 7
6.5 Semantic 7
6.6 Pragmatic 7
7 Deep learning 7
8 Deep learning methods 8
8.1 Deep Reinforcement Learning 8
8.2 Recurrent Neural Networks(RNNs) 8
8.3 Convolutional Neural Networks (CNN) 9
8.4 Boltzmann Machines 10
9 Automatic Text Summarization 10
10 Challenges in Automatic Text Summarization 11
10.1 Evaluation 11
10.2 Important sentence selection 11
10.3 Interpretability 11
10.4 Interpreting long sentences and jargons 12
10.5 Anaphora problem 12
10.6 Retaining the quality of the text 12
10.7 Word sense ambiguity 12
10.8 Meaningful, intuitive, and robust 12

10.9 Higher level of abstraction . . . 12
11 Types of Automatic Text Summarization . . . 12
11.1 Abstractive vs. extractive . . . 13
11.2 Single vs. Multi-document Summarization . . . 13
11.3 Indicative vs. Informative . . . 13
12 Summarization evaluation . . . 13
12.1 Human evaluation . . . 13
12.2 Automatic evaluation . . . 14
12.2.1 Recall-Oriented Understudy for Gisting Evaluation (ROUGE) 14
12.2.2 METEOR . . . 15
12.2.3 SARI . . . 15
13 conclusion . . . 15

2 The State of the Art in Text Summarization approaches 16
1 Introduction . . . 16
2 Extractive text summarization methods . . . 16
2.1 Term Frequency-Inverse Document Frequency Method . . . 16
2.2 Cluster Based Method . . . 17
2.3 Text Summarization with Neural Network . . . 18
2.4 Text Summarization with Fuzzy Logic . . . 19
2.5 Graph based Method . . . 19
2.6 Latent Semantic Analysis Method . . . 20
2.7 Machine Learning approach . . . 21
2.8 Query based summarization . . . 22
3 Abstractive text summarization methods . . . 22
3.1 Structured Based Approach . . . 22
3.1.1 Tree Based Method . . . 22
3.1.2 Template Based Method . . . 23
3.1.3 Ontology Based Method . . . 24
3.1.4 Lead and Body Phrase Method . . . 24
3.1.5 Rule Based Method . . . 25
3.2 Semantic Based Approach . . . 25
3.2.1 Multimodal semantic model . . . 25
3.2.2 Information item based method . . . 25
3.2.3 Semantic Graph Based Method . . . 26
4 Conclusion . . . 26

3 Conception 27
1 Introduction . . . 27
2 Objectif of our system . . . 27
3 Characteristics of article pages . . . 27
4 The architecture of our system . . . 29
4.1 Text extraction . . . 31
4.1.1 Pdf to images . . . 31
4.1.2 Image to text . . . 31
4.2 Document preprocessing . . . 33
4.2.1 Enhancing Text Formatting in Two-Column article . . . 33
4.2.2 Header Extraction and Removal . . . 34

4.2.3 References Removal . . . 35
4.2.4 Bracket Removal for Cleaner References . . . 35
4.2.5 Page Number Removal . . . 35
4.2.6 Table and Figure Extraction and Removal . . . 36
4.2.7 Titles extraction . . . 36
4.2.8 Paragraph continuation . . . 36
4.3 Segmentation . . . 37
4.4 Paragraphs preprocessing . . . 38
4.4.1 Stopword Removal . . . 38
4.4.2 Lowercasing Sentences . . . 38
4.4.3 Unwanted Character Removal . . . 38
4.4.4 Stemming . . . 38
4.5 Term Frequency-Inverse Document Frequency . . . 39
4.5.1 Term Frequency (TF) . . . 40
4.5.2 Inverse Document Frequency (IDF) . . . 40
4.6 Cosine similarity . . . 41
4.7 Sorting sentences . . . 42
4.8 Summary generation . . . 42
5 Conclusion . . . 42

4 Implementation **43**
1 Introduction . . . 43
2 Environment . . . 43
3 Software environment . . . 43
3.1 Python . . . 43
3.2 Pycharm iDE . . . 44
4 libraries . . . 44
4.1 Math . . . 44
4.2 Numpy . . . 44
4.3 NTLK . . . 44
4.4 Costumetkinter . . . 44
4.5 Re . . . 45
4.6 Pdf2image . . . 45
4.7 Os . . . 45
4.8 Natsort . . . 45
4.9 PIL . . . 45
4.10 Pytesseract . . . 45
5 Corpus . . . 45
6 System overview . . . 46
7 Usage scenario . . . 47
8 Results and Discussion . . . 53
8.1 System evaluation . . . 53
8.2 Results analysis . . . 54
9 Conclusion . . . 55
General Conclusion . . . 56
Bibliography . . . 62

List of Figures

1.1 Levels of NLP 6
1.2 Deep Reinforcement Learning Architecture 8
1.3 Recurrent Neural Networks 9
1.4 Convolutional Neural Network (CNN) Architecture 9
1.5 Boltzmann Machines Architecture 10
1.6 Text summarization 11

3.1 The fundamental components of the article pages that used in our approach . . 28
3.2 The architecture of the proposed text summarization system. 30
3.3 Example of article page with two columns 32
3.4 Example of the extracted text in two columns page 33
3.5 Example of Enhanced Text Formatting in Two-Column pages. 34
3.6 Example of article heading removal 35
3.7 Example of elements extracted in the preprocessing phase 37
3.8 segmentation architecture 38
3.9 Example of applying the text preprocessing steps. 39
3.10 summary exemple 42

4.1 Interface presentation 46
4.2 Extracted text 47
4.3 An example of the preprocessed document text. 48
4.4 The extraction of main titles within the article. 48
4.5 The extraction of figure legends within the article. 49
4.6 The extraction of table captions within the article. 49
4.7 The extraction of article heading and sub-heading condition. 50
4.8 The extraction of article heading and sub-heading. 50
4.9 Automatic selection 51
4.10 Manual selection 51
4.11 TF-IDF results 52
4.12 Cosine similarity results 52
4.13 Summary generation 53

List of Tables

4.1 System Specifications 43
4.2 F score Rouge evaluation results 55

General Introduction

In the era of information overload, where vast amounts of textual data are generated every day, the need for efficient text summarization techniques has become paramount. Automatic text summarization aims to condense lengthy documents into concise and coherent summaries, enabling users to quickly grasp the key information contained within the original text. This dissertation focuses on one of the fundamental approaches to automatic text summarization: TF-IDF (Term Frequency-Inverse Document Frequency) combined with cosine similarity. TF-IDF is a statistical measure widely used in natural language processing to evaluate the importance of a term in a document or a collection of documents. On the other hand, cosine similarity is a metric that measures the similarity between two non-zero vectors in a multi-dimensional space.

The main objective of this research is to develop a robust and effective text summarization system by employing TF-IDF and cosine similarity techniques. By leveraging the strengths of these methods, we aim to overcome the limitations of traditional extractive summarization techniques and produce high-quality summaries that capture the essence of the original text.

In this dissertation, we will explore the theoretical foundations of TF-IDF and cosine similarity, providing a comprehensive overview of their underlying principles and mathematical formulations. We will also discuss their relevance and applicability in the context of automatic text summarization.
Additionally, we will investigate various strategies for preprocessing textual data, including tokenization, stop-word removal, stemming, and other techniques aimed at improving the quality of the summarization process.
We will analyze the impact of these preprocessing steps on the performance of the summarization system and provide recommendations for optimal configuration. Furthermore, we will delve into the implementation details of the proposed system, describing the architecture, algorithms, and methodologies employed. We will highlight the key components and processes involved in the summarization pipeline, such as document representation, term weighting, similarity computation, and summary generation.

To evaluate the effectiveness of the developed system, we will conduct extensive experiments using diverse datasets and employ well-established evaluation metrics such as ROUGE (Recall-Oriented Understudy for Gisting Evaluation) to measure the quality of the generated summaries. The experimental results will be thoroughly analyzed and compared with existing state-of-the-art approaches to demonstrate the superiority and effectiveness of the proposed TF-IDF and cosine similarity-based text summarization system.

This dissertation aims to contribute to the field of automatic text summarization by harnessing the power of TF-IDF and cosine similarity techniques. Through the exploration of theoretical foundations, implementation details, and rigorous experimentation, we aspire to develop a robust and efficient summarization system capable of producing high-quality summaries that

accurately capture the essential information from the source texts. The findings of this research hold significant implications for numerous applications where text summarization plays a crucial role in facilitating information extraction and decision-making processes.

We have organized our work into four chapters, each focusing on a specific aspect of text summarization:

— Chapter 1: In this chapter, we provide a comprehensive study of various text summarization techniques.

— Chapter 2: Building upon the foundation laid in the previous chapter, we present an overview of recent studies and advancements in the field of text summarization.

— Chapter 3: This chapter is dedicated to the design and detailed architecture of our proposed text summarization system.

— Chapter 4: In this final chapter, we present the main results obtained through our research and provide interpretations and analyses.

Chapter 1

Automatic Text Summarization

1 Introduction

Natural Language Processing (NLP) has made significant progress in recent years, allowing computers to process and analyze human language in a way that was once thought impossible. One of the most popular applications of NLP is automatic text summarization, which involves generating a shorter version of a longer text while preserving its essential meaning.

Automatic text summarization has many practical applications, from helping users quickly grasp the main points of an article to enabling search engines to display relevant information in a more concise manner.

This chapter will explore the field of NLP and its applications. We will discuss the challenges involved in automatic text summarization, as well as the types and the techniques of evaluation. Overall, this chapter will provide an overview of NLP and automatic text summarization, demonstrating their importance and potential for the future.

2 Natural Language Processing

Natural Language Processing (NLP) is the study of language data using computational methods, often in the form of text such as documents or publications. The aim of NLP is to provide structure to unstructured natural language by leveraging insights from linguistics. This structure can be either syntactic, capturing the grammatical relationships within text, or semantic, representing the meaning conveyed. NLP is applied in systems biology to create applications that merge information extracted from literature with other biological data sources .[Verspoor et Cohen, 2013]

3 Objectives of NLP

In Natural Language Processing (NLP), one of the primary goals is to develop software or computer programs that can automatically process linguistic data. To achieve this objective, the first step is to make the rules of language explicit, which involves identifying and defining the various components of language, such as syntax, semantics, and pragmatics. The next step is to represent these rules in operational and calculable formalisms, which involves designing algorithms and models that can manipulate and process linguistic data. Finally, these rules and models are implemented using computer programs, which can process large volumes of linguistic data and produce useful outputs such as summaries, translations, or answers to questions.

This short-term objective is critical for advancing the field of NLP and creating applications that can benefit businesses, governments, and individuals.[MOHAMMED CHIKOUCHE, 2017]

Another goal of nlp is to verify linguistic theories and gain a deeper understanding of how humans communicate through language. To achieve this, NLP researchers use computers to simulate the human abilities of language comprehension and production. These simulations are then compared to human performance, allowing linguistic theories to be validated or improved.[MOHAMMED CHIKOUCHE, 2017]

4 Tools of NLP

Natural Language Processing (NLP) requires the use of tools from both formal linguistics and computer science:

4.1 Linguistic tools

Linguistic tools are concerned with describing various aspects of natural language, such as its vocabulary, grammar, syntax, and semantics. These tools are typically based on linguistic theories and research, and they aim to model how humans use language to communicate. Some examples of linguistic tools include lexicons, grammars, and ontologies.[BERROUBI *et al.*, 2022]

4.2 Formal tools

Formal tools are designed to express linguistic knowledge in a formalism that is suitable for automated processing[BERROUBI *et al.*, 2022]. This involves translating linguistic concepts and rules into mathematical or logical expressions that can be manipulated by a computer. Formal tools can include things like regular expressions, context-free grammars, and formal semantics. These tools allow NLP systems to process natural language text and extract meaning from it.[BERROUBI *et al.*, 2022]

4.3 Computational tools

Computational tools are concerned with implementing the formal descriptions of linguistic knowledge in actual computer programs[BERROUBI *et al.*, 2022]. These programs can perform a variety of NLP tasks, such as parsing sentences, generating text, and performing information retrieval. Computational tools can range from simple scripts to complex machine learning models, depending on the specific task at hand. Some examples of computational tools include part-of-speech taggers, named entity recognizers, and sentiment analysis classifiers.

5 Application of NLP

There are numerous applications of Natural Language Processing (NLP), which is a field of artificial intelligence that focuses on the interaction between computers and human language. Some of the most common applications of NLP include:

5.1 Machine Translation

Machine Translation (MT) is the process of automatically translating text from one language to another. One of the main challenges in MT is that there are many languages with different sentence structures and grammar rules. The goal of MT is not simply to translate words from one language to another, but to accurately capture the meaning of sentences while also preserving the correct grammar and tense. This is typically done using statistical models and machine learning algorithms that analyze large amounts of data to identify patterns and relationships between languages. Despite significant progress in MT over the years, it is still a challenging task, particularly for languages with complex syntax or for translating idiomatic expressions.[Khurana *et al.*, 2023]

5.2 Text Categorization

Text categorization or text classification is a popular application of natural language processing that involves assigning predefined categories or labels to large volumes of data, such as news articles, official documents, and market data. Categorization systems are used to save time and automate tasks that would otherwise require human indexers or staff to perform.[Khurana *et al.*, 2023]

5.3 Summarization

Summarization is a powerful application of NLP that utilizes the discourse level of analysis to condense larger texts into shorter, but still information-rich summaries that capture the essence of the original document.[Liddy, 2001]

5.4 Information Extraction

Information Extraction is a relatively new application area in NLP that is concerned with identifying and tagging specific pieces of information, such as names of people, companies, locations, and organizations, within large collections of text. This extracted information can be used for various purposes, such as question-answering, data visualization, and data mining. [Liddy, 2001]

5.5 Corrector

In general, language analysis tools such as spell checkers, grammar checkers, and style checkers are commonly used to help humans transform a text into a corrected version. These tools are widely popular and are typically included as a standard feature in most word processing software.[MOHAMMED CHIKOUCHE, 2017]

6 Levels of NLP

Natural Language Processing (NLP) is the field of study that deals with the interaction between computers and human language. NLP can be divided into various levels depending on the complexity of the task and the depth of the analysis required. See figure 1.1

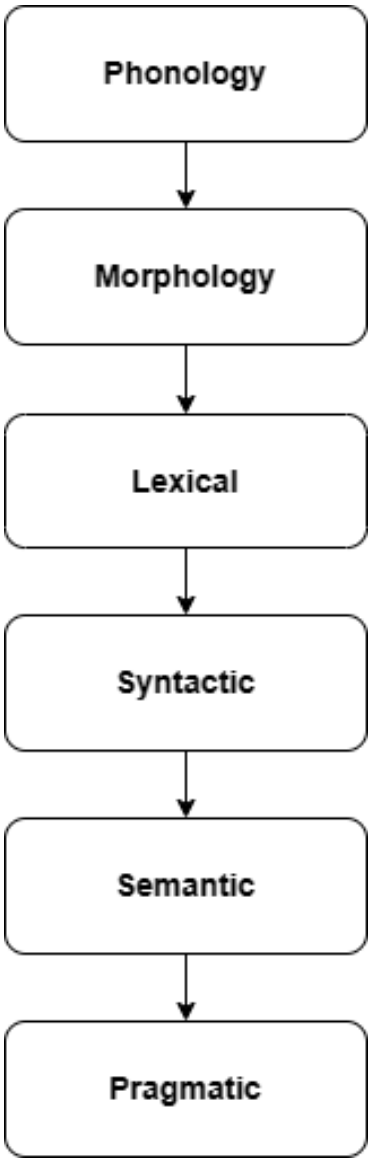

Figure 1.1: Levels of NLP
[MOHAMMED CHIKOUCHE, 2017]

6.1 Phonology

This level of NLP focuses on understanding speech sounds within and between words. Phonological analysis involves three types of rules:

1) phonetic rules for sounds within words.

2) phonemic rules for variations in pronunciation when words are spoken together.

3) prosodic rules for changes in stress and intonation throughout a sentence.

When an NLP system is designed to accept spoken input, the sound waves are analyzed and converted into a digitized signal for interpretation using various rules or by comparing them to a specific language model[Liddy, 2001].

6.2 Morphology

level of NLP deals with the smallest units of meaning in words called morphemes. Words are composed of these morphemes, and by breaking down an unknown word into its constituent morphemes, humans can understand its meaning. An example is the word preregistration, which can be morphologically analyzed into three separate morphemes: the prefix pre, the root registra, and the suffix tion. NLP systems can also recognize the meaning conveyed by each morpheme to gain and represent meaning. For instance, adding the suffix -ed to a verb indicates that the action took place in the past, which is a crucial piece of meaning that is often only conveyed through the use of the -ed morpheme in a text [Liddy, 2001].

6.3 Lexical

Both humans and NLP systems interpret the meanings of individual words. This involves various types of processing, with the first being the assignment of a single part-of-speech tag to each word. Words that can function as multiple parts of speech are assigned the most likely tag based on the context in which they appear [Liddy, 2001].

6.4 Syntactic

Syntactic processing involves analyzing sentence structure using a grammar and parser to identify the relationships between words. This process reveals how words depend on each other and contributes to the meaning of the sentence. Different grammars and parsers can be used, and not all NLP applications require a full parse of sentences. However, syntax is essential in conveying meaning in most languages because the order and dependency of words affect the meaning of a sentence. For example, "The dog chased the cat." and "The cat chased the dog." have different meanings due to syntax [Liddy, 2001].

6.5 Semantic

Semantic processing determines the possible meanings of a sentence by analyzing the interactions among word-level meanings in the sentence. It includes the semantic disambiguation of words with multiple senses, allowing only one sense of polysemous words to be included in the semantic representation of the sentence. Different methods can be used for disambiguation, including consideration of frequency, local context, and pragmatic knowledge [Liddy, 2001].

6.6 Pragmatic

The pragmatic level of language processing involves using context to understand the purposeful use of language in situations. It requires world knowledge, including the understanding of intentions, plans, and goals. Some NLP applications use knowledge bases and inferencing modules. An example of pragmatic language processing is resolving anaphoric terms like "they" based on world knowledge [Liddy, 2001].

7 Deep learning

Deep learning is a set of techniques in machine learning that have made significant advances in artificial intelligence in recent years. In machine learning, a program analyzes a data set to

draw rules that will allow it to make conclusions about new data. Deep learning is based on what has been called, by analogy, "artificial neural networks," made up of thousands of units (the "neurons") that each perform simple operations. The results from one layer of neurons are used as inputs for the calculations of another layer, and so on. For example, in visual recognition, early layer units identify lines, curves, angles, etc., while higher layers identify shapes, combinations of shapes, objects, and contexts. The progress in deep learning has been made possible by the increase in computer power and the development of large data bases (big data).[Boughaba *et al.*, 2017]

8 Deep learning methods

Deep learning is a subfield of machine learning that uses deep artificial neural networks to model complex relationships and patterns in large amounts of data. Some popular deep learning methods include:

8.1 Deep Reinforcement Learning

Reinforcement learning is a process where an agent interacts with its environment to change its state. The agent takes actions based on observations of its surroundings in order to achieve a goal. The network architecture consists of an input layer (the state of the environment), an output layer, and multiple hidden layers. The principle of reinforcement learning is based on repeated predictions of the future payoff for each action taken in a particular situation. See figure 1.2[Divya et Aiswarya, 2021]

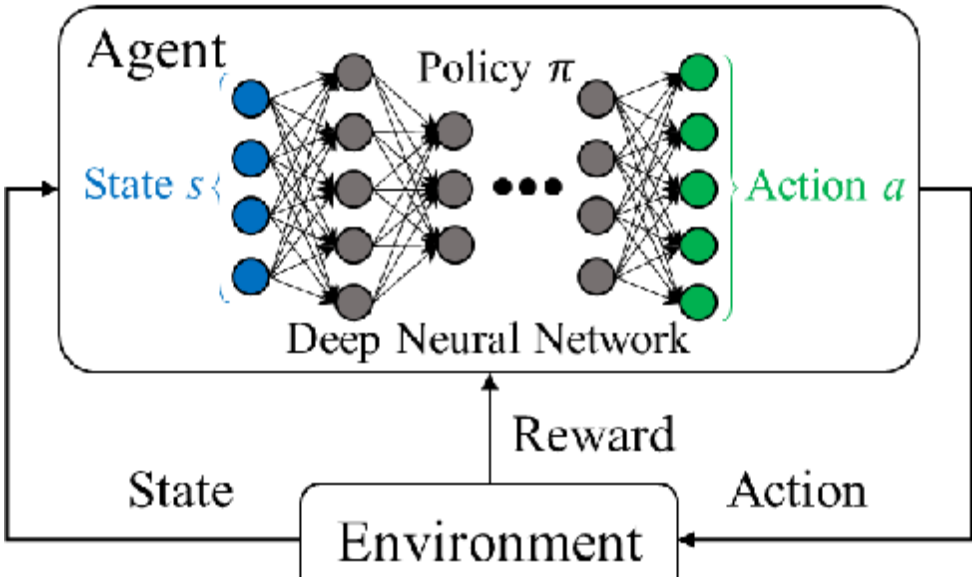

Figure 1.2: Deep Reinforcement Learning Architecture
[Huang *et al.*, 2019]

8.2 Recurrent Neural Networks(RNNs)

A Recurrent Neural Network (RNN) is a form of Neural Network (NN) that specializes in handling sequential or time-series data. It is commonly used in fields such as machine translation, speech recognition, DNA sequencing, and modeling of system dynamics. RNNs have a basic

structure that consists of an input layer, one or more hidden layers, and an output layer. See figure 1.3[Abdelrahman, 2019]

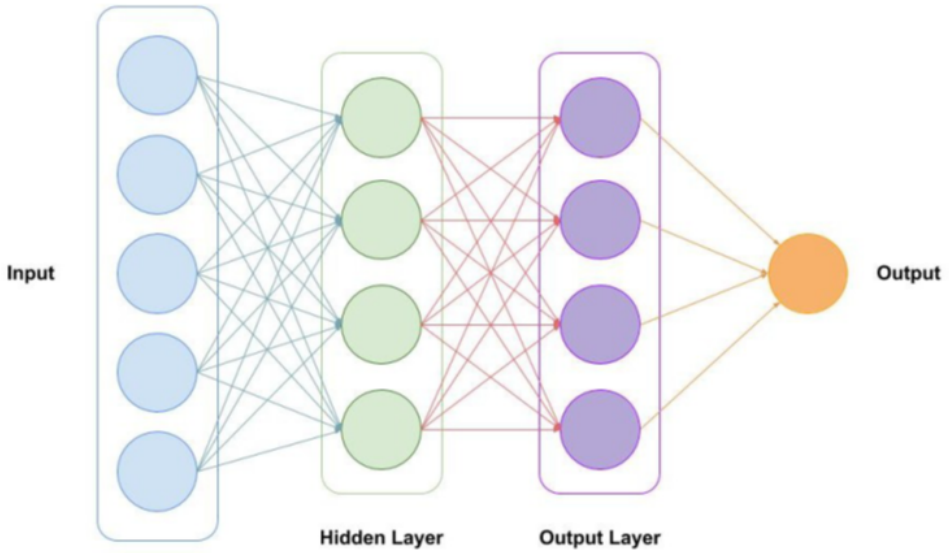

Figure 1.3: Recurrent Neural Networks

8.3 Convolutional Neural Networks (CNN)

A Convolutional Neural Network (CNN) is a specialized form of artificial neural network designed specifically for analyzing pixel data in image recognition and processing. As an AI system that uses deep learning, CNNs can perform both generative and descriptive tasks, such as image and video recognition, recommender systems, and natural language processing. A neural network is a system, in either hardware or software, modeled after the functioning of neurons in the human brain. Traditional neural networks are not equipped for image processing and must break down pictures into smaller parts for processing. See figure 1.4[Divya et Aiswarya, 2021]

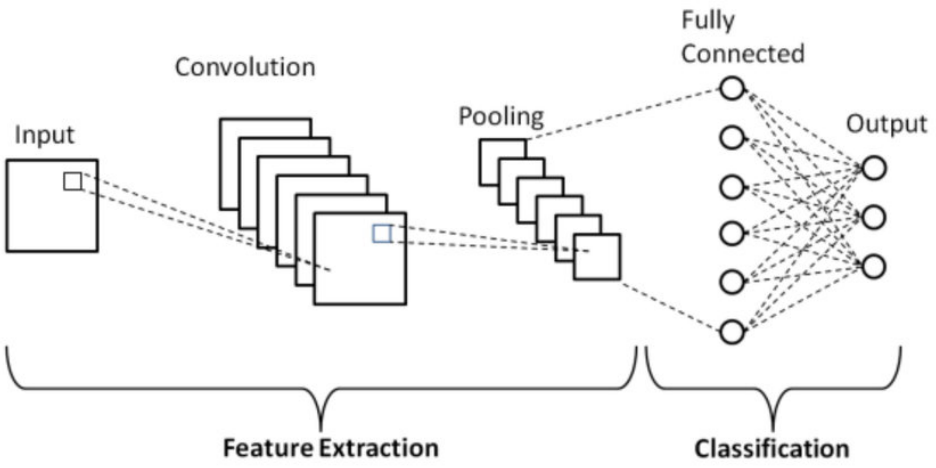

Figure 1.4: Convolutional Neural Network (CNN) Architecture
[Ray *et al.*, 2020]

8.4 Boltzmann Machines

A Boltzmann Machine is an artificial neural network composed of interconnected, binary units that make random decisions about their state of being "on" or "off." These units are symmetrically connected, and the network can be trained to identify features in datasets made up of binary vectors. Although the learning algorithm for Boltzmann machines can be slow for networks with many layers, it can be accelerated by learning one layer at a time. Boltzmann machines have two distinct uses in computation. In a search problem, the connections' weights are fixed to represent the cost function of an optimization problem, and the stochastic dynamics of the Boltzmann machine enable it to generate binary state vectors that are good solutions to the optimization. In a learning problem, the machine is shown a set of binary data vectors and must find weights that result in the data vectors being good solutions to the optimization problem defined by those weights. To solve a learning problem, the Boltzmann machine makes multiple, small updates to its weights, which requires it to solve numerous search problems. See figure 1.5[Hinton, 2007]

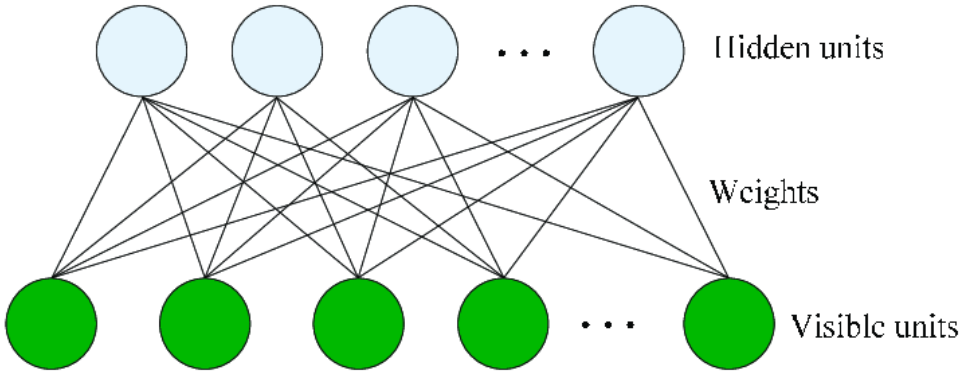

Figure 1.5: Boltzmann Machines Architecture
[Li *et al.*, 2018]

9 Automatic Text Summarization

Automatic text summarization generates a shortened version of a given document by extracting its most meaningful information and writing it using human language. It can be performed on a single document or multiple documents, with single document summarization having less redundancy and time-related issues. The summarization can also be generic or query-focused, adapting to the reader's topic of interest. The summarization can be extractive, using sentences from the source text, or abstractive, generating novel sentences for a more natural summary that only includes relevant information.[Romero, 2017]

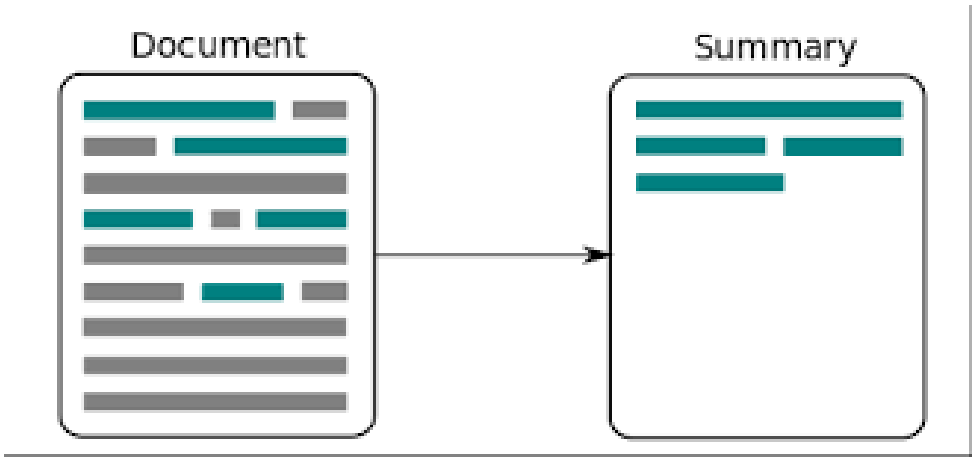

Figure 1.6: Text summarization
[Yadav *et al.*, 2022]

10 Challenges in Automatic Text Summarization

Automatic text summarization is a challenging task in the field of natural language processing due to the complexity and variability of human language. Summarizing text requires a deep understanding of the content and structure of the text, as well as the ability to identify and condense the most important information. Some of the main challenges of automatic text summarization include:

10.1 Evaluation

For automated evaluation techniques such as ROUGE and BLEU, the scores generated for extractive summarization tend to have less relevance than those obtained through human evaluation. Furthermore, their limited vocabulary can make it difficult to find synonyms for words used in documents, and they require a large number of reference sentences to work effectively. Additionally, grammatically and semantically incorrect sentences are often overlooked, leading to unreliable scores, with some metrics giving high marks to trivial sentences while failing to penalize incorrect grammar.[Mridha *et al.*, 2021]

10.2 Important sentence selection

Typically, an ATS system chooses the most pertinent sentences from the original text and designates them as crucial. However, when forming the summary, it's necessary to standardize the selection of sentences or words based on established criteria. The challenge is that determining the significance of sentences is highly subjective. To overcome this, custom data can be utilized to produce professional summaries. While vector representation and similarity matrices try to identify word correlations, there is currently no foolproof method for determining the most critical sentences.[Mridha *et al.*, 2021]

10.3 Interpretability

Abstractive models offer condensed versions of source material that capture its central ideas. However, due to the complexities of human language and expression of emotions in written

works, machines often struggle with effectively interpreting source content through abstract models. This makes it a challenging task to ensure the comprehensibility of source content in abstract models.[Mridha *et al.*, 2021]

10.4 Interpreting long sentences and jargons

Existing learning methods often have difficulty summarizing lengthy sentences, causing confusion for algorithms during source text processing. To resolve this challenge, it's necessary for researchers to identify the problem and create new systems that minimize or eliminate this problem.[Mridha *et al.*, 2021]

10.5 Anaphora problem

The "anaphora problem" is a common challenge in text summarization, where individuals often use synonyms or pronouns to refer to the subject during a discussion. This issue involves determining which pronoun corresponds to which word.[Mridha *et al.*, 2021]

10.6 Retaining the quality of the text

An automatic text summarization system should aim to maintain the quality of the summarized content. From the user's perspective, the most important quality of ATS is the ability to comprehend the source text during the summarization process. Various machine learning techniques can be utilized to preserve the quality of the summarized text.[Mridha *et al.*, 2021]

10.7 Word sense ambiguity

The ambiguity of words can impact the summarization of sentences. This ambiguity can occur due to the use of abbreviations with multiple meanings, multiple uses of the same word in different contexts, etc. To improve comprehension, the acronym must align with the topic or context based on the subject. This issue, known as the cataphora problem, is the reverse of the anaphora problem. It can be addressed using a disambiguation algorithm.[Mridha *et al.*, 2021]

10.8 Meaningful, intuitive, and robust

The sentences in a summary must be impactful and meaningful to the users and have a clear representation, even in the face of difficulties faced by the system.[Mridha *et al.*, 2021]

10.9 Higher level of abstraction

The pursuit of higher-level abstraction in text summarization remains an open research topic, presenting ample opportunities for researchers and linguists to find solutions.[Mridha *et al.*, 2021]

11 Types of Automatic Text Summarization

There are several types of automatic text summarization, including:

11.1 Abstractive vs. extractive

There are two main techniques for automatic summarization: extractive and abstractive. The extractive method involves extracting important sentences or phrases from the source text without modifications, and combining them to create the summary (Gupta and Lehal, 2010). The goal of extractive summarization is to identify important sentences and generate a shortened version that accurately represents the original text. The abstractive method, on the other hand, uses linguistic techniques for a deeper analysis of the text to generate a new and different representation. Abstractive summaries are usually similar to human-written ones, as they represent the content and meaning of the original text in a more natural way. Although abstractive summarization is more efficient, its implementation requires a deep understanding of deep learning techniques, so most researchers tend to focus on extractive summarization.[Kantzola, 2020]

11.2 Single vs. Multi-document Summarization

can be generated in two forms depending on user requirements: single-document and multi-document summarization. Single-document summarization processes and summarizes information from one source document, while multi-document summarization combines information from multiple related documents into one summary. The same techniques can be applied to both types of summarization, but summarizing multiple documents presents greater challenges in terms of coherence as the summary needs to represent the overall meaning of all documents in a compressed form. Query relevant summarization refers to summaries generated based on a query or question and is related to information retrieval.[Kantzola, 2020]

11.3 Indicative vs. Informative

Summarization can be categorized into two types based on the desired output: indicative and informative. Indicative summarization only includes the main idea of the source text and gives a brief overview, leaving the decision of reading the full text to the reader. Typically, indicative summaries are 5-10 % of the original document. On the other hand, informative summarization includes all important information from the source text but in a concise format, serving as a substitute for the full text. Informative summaries are usually 20-30 % of the original document.[Kantzola, 2020]

12 Summarization evaluation

Human evaluation and automatic evaluation are two approaches used to evaluate the quality of generated outputs, such as summaries, translations, or answers to questions.

12.1 Human evaluation

Evaluation of summaries by humans typically involves creating a set of questions for a group of raters to answer. These questions can assess qualitative features like fluency or grammar, or evaluate how much of the original meaning is retained in the summary. Ratings are commonly given on a numerical scale of 1 to 5, although it is also common to ask subjects to choose their preferred summary.[Nikolov, 2020]

12.2 Automatic evaluation

Summarization evaluation involves assessing the quality of automatically generated summaries. It helps determine the effectiveness of summarization algorithms. Different metrics, such as ROUGE, BLEU, and METEOR are used to evaluate the generated summaries based on factors such as word overlap and semantic similarity with reference summaries. The selection of the appropriate evaluation metric and reference summary can be complex, but summarization evaluation is still essential for improving automatic summarization.

12.2.1 Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

ROUGE, a recall-based metric, is widely used to evaluate summarization due to its strong correlation with human assessments. The two most frequently used variants in the literature are ROUGE-N and ROUGE-L.

ROUGE-N:N-gram Co-Occurrence Statistics: measures the overlapping of N-grams between the system summary and the gold standard summary, expressed as a ratio of the number of co-occurring N-grams to the total number of N-grams in the gold standard. [Lin, 2004a]

ROUGE-L: Longest Common Subsequence In the field of automatic summarization evaluation, the longest common subsequence (LCS) is used to compare the similarity between two texts. The LCS of two sequences X and Y is a common subsequence with the maximum length. To apply this method in summarization evaluation, the authors Saggion et al.[Saggion *et al.*, 2002] used normalized pairwise LCS.

The LCS-based metric, known as ROUGE-L, calculates the ratio between the length of the LCS of the two summaries and the length of the reference summary. This method views a summary sentence as a sequence of words. One advantage of using LCS is that it does not require consecutive matches, but rather in-sequence matches that reflect sentence-level word order. Another advantage is that it automatically includes the longest in-sequence common n-grams, so there is no need to specify a predefined n-gram length. [Lin et Och, 2004]

ROUGE-W: Weighted Longest Common Subsequence The traditional LCS algorithm has a limitation in that it doesn't take into account the different spatial relationships within a sequence. To overcome this issue, they propose a new metric called ROUGE-W or the weighted longest common subsequence. This metric gives higher priority to LCSes that have consecutive matches. ROUGE-W can be quickly calculated using a dynamic programming approach [Lin et Och, 2004][Lin, 2004a].

ROUGE-S: Skip-Bigram CoOccurrence Statistics ROUGE-S is used to compare a generated summary to a reference summary, and it calculates the overlap between the two in terms of unigrams (single words). The score is expressed as a percentage, with higher scores indicating greater similarity between the two summaries. For example, sentence "police killed the gunman" has C(4,2)4 = 6 skip-bigrams: ("police killed", "police the", "police gunman", "killed the", "killed gunman", "the gunman").[Lin et Och, 2004]

ROUGE-SU: Extension of ROUGE-S ROUGE-S may not accurately evaluate a candidate sentence if it does not contain any word pairs that co-occur with the reference summaries. To address this issue, ROUGE-S has been extended to include unigram as a counting unit. This

extended version is known as ROUGE-SU. The effectiveness of various forms of these ROUGE metrics will be assessed in the following section using three years of DUC data.[Lin et Och, 2004]

12.2.2 METEOR

METEOR is a commonly used precision-based evaluation metric for MT. It aligns system-generated sentences to a gold standard and calculates similarity scores based on word/phrase matching (exact, stem, synonym, or paraphrase). METEOR has shown better correlation with human translators than BLEU and is used in automatic summarization studies for a different evaluation perspective than ROUGE.[Papineni *et al.*, 2002]

12.2.3 SARI

SARI is a recent metric that evaluates the quality of summaries by considering the total number of additions and deletions from the original text. Initially developed for sentence simplification, SARI can also be applied to summarize text.[Xu *et al.*, 2016]

13 conclusion

In conclusion, this chapter has discussed the field of Natural Language Processing (NLP), including its goals, tools, levels and applications, specifically in automatic text summarization. We have explored the challenges associated with this task, which include understanding the meaning and context of the original text and generating a summary that is both accurate and easy to read.

We also explored the types of automatic text summarization and also the types and techniques of evaluation used to measure the effectiveness of automatic text summarization.

Chapter 2

The State of the Art in Text Summarization approaches

1 Introduction

Text summarization is the process of creating a shorter version of a longer text while preserving its key information. There are two primary methods of text summarization: extractive and abstractive.

In this chapter, we will explore the different techniques used in abstractive and extractive text summarization. We will examine the strengths and weaknesses of each method and provide examples of their use in various applications, such as news summarization, document summarization, and social media summarization.

2 Extractive text summarization methods

Extractive summarization involves selecting important sentences or phrases from the original text and using them to create a summary. This method is straightforward and commonly used in news articles or other texts with a clear structure. Extractive summarization algorithms typically rely on statistical and machine learning techniques to identify the most relevant sentences.

2.1 Term Frequency-Inverse Document Frequency Method

Term frequency (TF) and inverse document frequency (IDF) are numerical measures that indicate the importance of a word in a given document. TF is the count of how many times a term appears in the document, while IDF reduces the weight of frequently occurring terms and increases the weight of rare terms. Sentences are then scored based on this calculation, with those receiving high scores being included in the summary. However, a potential issue with this method is that longer sentences may receive higher scores simply because they contain more words.[Andhale et Bewoor, 2016]

Arunlfo and Ledeneva's [García-Hernández et Ledeneva, 2009]approach uses tf-idf to calculate word importance in a given document. They then employ an unsupervised learning algorithm to create a non-redundant summary that includes only the most critical information from

the source material. The approach weighs the most significant terms in the document using tf-idf, which considers both the term frequency within the document and the inverse document frequency across the entire collection of documents. This mitigates the impact of common words in the document or collection and gives more significance to rare words that may contain more relevant information. Using an unsupervised learning algorithm enables the creation of a summary without prior knowledge of what to include or exclude, producing a comprehensive and accurate summary.

[Sarkar, 2012] proposed a new approach to news summarization that addresses some of the issues with the traditional tf-idf method. The problem with tf-idf is that longer sentences can sometimes get higher scores simply because they contain more words, which may not necessarily make them more important for summarization.
Sarkar proposed an approach by incorporating sentence features in addition to tf-idf scores. These features include sentence length, noun and verb count, and quotes or named entities. The combination of these features with tf-idf scores produces more accurate summaries that focus on essential information. However, determining optimal weights for each feature and manual selection of important features remain a challenge.

[Baralis *et al.*, 2015]proposed a method for document summarization based on a weighted item set model. This approach involves first identifying and connecting various significant terms in the document, and then assigning weights to these terms using the tf-idf method. The weights are used to extract related item sets from the document, which are then used to generate a summary. The idea behind this approach is that by identifying significant terms and their relationships within the document, the summary can capture the most important information while avoiding redundancy. The use of the tf-idf weighting scheme ensures that rare and important terms are given higher weights, while common and less important terms are given lower weights.

[Jayashree *et al.*, 2012] proposed a keyword extraction approach using TF-IDF that calculates the frequency of each term in the document and multiplies it with IDF score to obtain a weighted score for each term. GSS, or the Greedy Set Cover algorithm, is a probabilistic feature selection method used in combination with TF-IDF to identify the most important words for summary. It iteratively adds words to the summary until the desired length is reached or no more words can be added without exceeding the length constraint, using a combination of TF, IDF, and coverage score to calculate the score for each word. This approach can effectively identify the most informative words for summary.

2.2 Cluster Based Method

[Zhang et Li, 2009]proposed a method to automatically generate summaries of documents using a clustering approach. Specifically, they used the K-means clustering algorithm to group similar sentences together into clusters. The sentence features used to define similarity between sentences could be various, such as word frequency, length, or syntactic structure.
Once the clusters were formed, the central sentence of each cluster was selected as the representative summary sentence for that cluster. This was based on the assumption that the central sentence would capture the main idea of the cluster, and therefore would be an effective summary.

The advantage of this approach is that it is unsupervised, meaning it does not require human-annotated data to train a model. However, the quality of the summaries generated can be affected by the choice of features and the clustering algorithm used.

The approach proposed by [Wu *et al.*, 2015] combines spectral clustering and the LexRank algorithm for summarization. In the first step, the k-nearest neighbor method generates a sparse matrix of similar sentences, which is then used for spectral clustering to group similar sentences based on their content and context. In the second step, the LexRank algorithm is applied to the clusters to select the most representative sentences for the summary based on their similarity scores. This approach generates summaries that contain all the important information from the document while avoiding redundancy, and has been shown to be effective in generating high-quality summaries for various types of documents.

[Ferreira *et al.*, 2014] proposed a summarization algorithm that converts the document into a graph representation, with each sentence as a node and edges representing sentence similarity. TextRank is then used to identify important sentences based on their connections in the graph. The algorithm then clusters similar sentences to create groups that represent different topics, and selects the most representative sentence from each cluster for the summary. This approach effectively identifies important information and themes in the document, resulting in high-quality summaries.

2.3 Text Summarization with Neural Network

The article [Kaikhah, 2004] introduces an innovative approach to automatic text summarization using a neural network. By training the network on a set of example texts, it acquires the ability to recognize the most important features for summarization, such as significant keywords, relevant phrases, and key concepts. During the summarization process, each sentence in the input text is assigned a score based on its feature vector, which represents the presence or absence of these crucial features. To streamline the model and improve efficiency, the neural network undergoes pruning to eliminate uncommon features while preserving the essential ones. The final summary is generated by selecting sentences with the highest scores, ensuring that the most pertinent and informative information is conveyed. This method has the potential to produce concise and accurate summaries, but it necessitates meticulous feature selection, representation, and high-quality training data to achieve optimal results.

[Thu *et al.*, 2013]proposed a Vietnamese text summarization method based on neural networks,which is designed to reduce computation and improve performance. Their approach utilizes semi-supervised learning, which improves the summary quality by learning from both labeled and unlabeled data. The sentences in the document are scored based on the words they contain. A trained neural network predicts the sentence importance, and the highest-scoring sentences are selected for the summary. This method is efficient, produces high-quality summaries, and reduces the need for labeled data.

[Kågebäck *et al.*, 2014] proposed a summarization method that utilizes an autoencoder to derive phrase embeddings. In this approach, the input document is first parsed using a recursive neural network to generate a binary parse tree. The phrase embeddings are then derived by summing the word embeddings for each phrase in the parse tree.
Once the phrase embeddings have been derived, summarization is performed by measuring the

similarity between phrases. The most similar phrases are selected for the summary. The system also takes into account the length of the summary, ensuring that the summary is concise and contains only the most important information.

2.4 Text Summarization with Fuzzy Logic

[Babar et Patil, 2015]proposed a summarization method that utilizes fuzzy rules and triangular membership functions to score sentences based on their features. In this approach, the features of each sentence are identified and scored using the fuzzy rules and triangular membership functions. These scores are then used to rank the sentences in the document.
To further improve the quality of the summary, the system also employs latent semantic analysis. This technique is used to identify the underlying concepts and themes in the document, which can help to improve the coherence and relevance of the summary.
Overall, this approach has the advantage of being able to handle uncertainty and ambiguity in the scoring process, allowing for more accurate and reliable summarization results.

[Hannah *et al.*, 2011] proposed a summarization method that employs a fuzzy inference system for scoring sentences based on their features. The fuzzy inference system identifies and scores the features of each sentence, which are then aggregated to generate an overall score. This score is used to rank the sentences and select the most relevant ones for the summary. The fuzzy inference system handles uncertainty and ambiguity in the scoring process, resulting in accurate and coherent summaries. The approach has been effective in summarizing various document types, including scientific papers and news articles.

[Modaresi et Conrad, 2014] proposed a summarization method that uses a fuzzy set of phrases to summarize documents. Key phrases are identified based on high membership values in the fuzzy set, determined using the maximum entropy model. The method also considers sentence features to rank sentences, selecting those with higher membership values as more relevant and informative for the summary. This approach effectively handles uncertainty and ambiguity, producing accurate and reliable summaries. It has been successful in summarizing various document types, including scientific papers and news articles. (Modaresi, 2014)

2.5 Graph based Method

[Mihalcea, 2004] investigated different ranking algorithms for summarization. In her approach, the similarity between sentences is determined based on token overlap. The more tokens two sentences share, the more similar they are considered to be. This approach is used to calculate a similarity score between each pair of sentences in the document.
The sentences are then ranked based on their similarity scores, with the most similar sentences appearing at the top of the list. The summary is generated by selecting the highest ranked sentences, with a limit on the total number of sentences to be included in the summary.
This approach has the advantage of being simple and easy to implement, while still producing reasonable summaries. However, it may not be as effective as more advanced methods in capturing the semantic meaning of the sentences, and may produce summaries that are less coherent or informative.

[Malliaros et Skianis, 2015] proposed a method that uses node centrality to indicate the importance of terms in a document. Specifically, they consider both local and global node centralities for term weighting.
Local node centrality is calculated based on the frequency of the term in the document, while global node centrality is calculated based on the degree of the term in the document's word co-occurrence network.
Using these centrality measures, each term is assigned a weight that reflects its importance in the document. Sentences are then scored based on the sum of the weights of the terms they contain, and the highest-scoring sentences are selected for the summary.
This approach has the advantage of being able to capture the importance of individual terms in the document, and can produce informative and coherent summaries. However, it may not be as effective as more advanced methods in capturing the overall meaning and structure of the document, and may produce summaries that are less concise or focused.

[Cheng *et al.*, 2013] proposed a summarization approach that builds a dependency graph using both term co-occurrence relations and syntactic relations. The authors then use a depth-first traversal to extract a subgraph of three nodes that is used to generate the summary.
The nodes in the subgraph are selected based on their importance in the document, which is determined by their degree and centrality in the graph. The authors also consider the position of the nodes in the graph and the frequency of their corresponding terms in the document.
Once the subgraph is constructed, the authors use a set of rules to generate a summary sentence that captures the key information in the subgraph. These rules take into account the syntactic structure of the sentence and the relationships between the nodes in the subgraph. This approach has the advantage of being able to capture both the semantic and syntactic structure of the document, which can lead to more informative and coherent summaries. However, it may require more computational resources and may not be as effective on very large documents or documents with complex structures.

2.6 Latent Semantic Analysis Method

In the method proposed by [Ozsoy *et al.*, 2011], each sentence in a document is represented as a row in a matrix, while each concept is represented as a column. The authors use an algorithm to fill the matrix with scores indicating the degree of association between each sentence and concept. This is done by computing the cosine similarity between the sentence and the concept, where the concept is represented as the centroid of the words that appear in it.
After the matrix is constructed, the authors propose two methods for sentence selection: the cross method and the topic method. The cross method selects the highest-scoring sentence from each row and column of the matrix, resulting in a summary that covers a broad range of topics. On the other hand, the topic method calculates the score of each concept based on the edge weights in the graph and selects the sentences with the highest concept scores. This method generates a summary that is more focused on specific topics.
.

[Geetha et Deepamala, 2015] proposed a method to convert a document into a sentence matrix where each word is represented by rows and each sentence is represented by columns. This method was initially proposed by Steinberger and Jezek [Steinberger *et al.*, 2004]. In this method, each cell of the matrix is filled with the product of the term frequency (TF) and inverse document frequency (IDF) values of the corresponding word in the sentence. In addition, the

Eagan value and Eagan vector are also calculated for each cell. The Eagan value represents the importance of a cell in the matrix and is used to weigh the sentence scores in the summary generation process.
The authors proposed two methods for sentence selection: the cross method and the average concept method. In the cross method, the sentence with the highest Eagan value in each row and column is selected to form a cross. The sentences that are part of this cross are then used to generate the summary. In the average concept method, the concepts are identified based on the Eagan vector and the average Eagan value of each concept is calculated. The sentences that contain these high-scoring concepts are then selected to generate the summary.

2.7 Machine Learning approach

In the approach proposed by [Thu et Ngoc, 2014], a Bayesian Network is used to identify important sentences from the document. The sentences are initially assigned a probability value based on their similarity to the document topic. Then, the probability values are updated based on the contextual information obtained from neighboring sentences.
The sentences are then ranked based on the probability difference between their updated and initial probabilities. The highest-ranked sentences are selected to form the summary. To further improve the efficiency of the method, a dynamic programming algorithm is used to find the weightiest path from the source node to the following nodes in the network. This reduces the computational cost of the summarization process. Overall, the approach aims to generate a high-quality summary while minimizing the computational resources required.

Bagging is a machine learning approach proposed by [Sarkar *et al.*, 2011] for text summarization, which uses decision trees as learners. In this approach, a bag of decision trees is trained with a sentence feature set. The feature set includes various linguistic features such as sentence length, word frequency, and part-of-speech tags. Each decision tree in the bag learns a different subset of features and creates its own classification model. During summarization, each sentence is classified by each decision tree, and the final decision is made by combining the results of all decision trees. The sentences that are classified as important by the majority of decision trees are included in the summary. This approach is effective in reducing the bias of a single decision tree and improving the accuracy of the summary.

Reinforcement learning is a type of machine learning where an agent learns to take actions based on a reward signal. In the approach proposed by [Prakash et Shukla, 2014], a term sentence matrix is first calculated for each term in the document using TF*IDF. Then, a sentence signature matrix is used to score sentences. The sentence signature matrix is calculated by first generating a vector for each sentence, where each element of the vector corresponds to the score of a term in the sentence. These scores are calculated using the term sentence matrix. The vectors for each sentence are then normalized and combined into a matrix.
Sentences are selected for the summary by using a reinforcement learning algorithm that takes the sentence signature matrix as input and outputs a sequence of actions (i.e., whether to include or exclude each sentence in the summary). The reward signal for the algorithm is based on the cosine angle between the sentence signature matrix and the summary signature matrix, which is calculated by combining the vectors of the selected sentences. The reinforcement learning algorithm learns to select sentences that maximize the cosine angle, and the resulting sequence of actions is used to generate the summary.

2.8 Query based summarization

[He *et al.*, 2008]proposed a feature fusion-based approach for summarizing text documents. The approach involves using both similarity and skip bigram co-occurrence features to calculate query relevance. The authors start by expanding the user query with high frequency words to avoid the problem of data sparseness. The similarity feature is calculated by considering the similarity between query and sentence embeddings, while the skip bigram co-occurrence feature is calculated by considering the frequency of skip bigrams in the sentences.
The authors then use a feature fusion strategy to combine the similarity and skip bigram co-occurrence features. The fusion strategy involves weighting the two features and using them to rank the sentences. The authors also proposed a sentence clustering method to group similar sentences together and generate a summary from each cluster. The authors also demonstrated the effectiveness of their approach in handling long documents.

[Varadarajan et Hristidis, 2006] proposed a method that uses the semantic associations between words within a document to construct a query-specific summary. The approach involves constructing a document graph, where each node represents a word or a phrase, and edges represent semantic associations between nodes. The graph is then traversed using a greedy approach to identify the content that is most relevant to the query, resulting in a more coherent summary. The approach is particularly useful when summarizing long and complex documents, where traditional summarization techniques may not capture the context and semantics of the document accurately.

3 Abstractive text summarization methods

Abstractive summarization, on the other hand, involves generating a summary that is not simply a copy of sentences from the original text, but rather a new text that conveys the most important information in a concise and coherent way. This approach is more complex than extractive summarization and requires natural language processing and machine learning techniques to generate summaries that are more similar to human writing.

3.1 Structured Based Approach

3.1.1 Tree Based Method

[Barzilay et McKeown, 2005] proposed a method for summarizing text that utilizes dependency trees and a theme insertion algorithm.
First, they used a shallow parser to parse the text and create dependency trees, which represent the relationships between words in a sentence. These dependency trees were then mapped to a predicate-argument structure, which captures the semantic roles of each word in the sentence.
Next, they used a theme insertion algorithm to identify the most important sentences in the text. This algorithm identifies "themes" in the text, which are concepts or entities that appear frequently across multiple sentences. Sentences that contain these themes are considered to be more important than other sentences, and are therefore selected for inclusion in the summary.
Finally, the summary sentences are generated using the high-ranking themes, and are expressed in the SURGE language, which is designed to be concise and readable.
Overall, this method takes advantage of both syntactic and semantic information to identify

important sentences and generate informative summaries. However, the accuracy of the method depends on the quality of the dependency trees and the effectiveness of the theme insertion algorithm.

[Bing *et al.*, 2015] presented a summarization method that focuses on identifying important concepts and facts in the original text. To achieve this, they first extract noun and verb phrases from the text using dependency trees, which represent the syntactic relationships between words in a sentence.
Once the noun and verb phrases have been identified, they are used to construct a constituent tree, which represents the structure of the text in terms of these phrases. The constituent tree can then be used to identify important concepts and facts in the text, based on their frequency and their position in the tree.
To generate a grammatically correct summary, they used Integer Linear Programming (ILP) to identify the most important phrases in the constituent tree and combine them into a coherent summary. ILP is a mathematical optimization technique that allows for the identification of the optimal combination of phrases, subject to various constraints such as length and grammaticality.
Overall, this method is effective in identifying important concepts and facts in the text, and generating grammatically correct summaries. However, the accuracy of the method depends on the quality of the constituent tree and the effectiveness of the ILP optimization.

3.1.2 Template Based Method

[Harabagiu et Lacatusu, 2002] proposed a summarization method that utilizes templates to extract relevant information from multiple documents.
The method involves creating an ad hoc template, which is a structured representation of the information that needs to be extracted from the documents. The template includes placeholders for relevant information, such as the name of a person, the date of an event, or the location of a place.
The template is then iteratively filled with snippets of text from multiple documents that follow the same pattern and rules defined in the template. The snippets are selected based on their relevance to the information being extracted, and their coherence with the other snippets that have been selected.
Finally, the filled template is used to generate a summary that includes the most important information extracted from the documents.
However, the effectiveness of the method depends on the quality of the templates and the accuracy of the rules used to fill them.

[Embar *et al.*, 2013] proposed a summarization system that uses an abstraction scheme and a domain template with Information Retrieval (IR) rules.
The abstraction scheme is a method of identifying and removing redundant or irrelevant information from the text, while preserving the most important information. The domain template is a set of predefined structures or patterns that are specific to the domain of the text being summarized. The IR rules are used to extract relevant information from the text based on the domain template.
The system creates a set of templates with a variety of forms that can be used to generate a summary. The template selection process involves selecting the template that best matches the abstraction scheme and the domain of the text being summarized. The selected template is then

filled with the relevant information extracted from the text using the IR rules.
The advantage of this approach is that it can be customized for different domains and can produce summaries in a variety of forms, depending on the needs of the user. However, the effectiveness of the method depends on the quality of the abstraction scheme, the domain template, and the accuracy of the IR rules used to extract information.

3.1.3 Ontology Based Method

[Lee *et al.*, 2005] proposed a fuzzy system for text summarization that uses an ontology designed by a News domain expert.
The system classifies sentences according to a term classifier, which is based on the domain ontology. The term classifier calculates the degree of relevance of each sentence to the ontology and assigns a membership degree to it. The fuzzy inference mechanism then uses the membership degree of each sentence to calculate a weight for it.
The weight of each sentence is used to determine its importance in the summary. The system selects the top-ranked sentences based on their weights to generate the summary. The advantage of this approach is that it can handle the ambiguity and uncertainty in the text by using fuzzy logic.

[Ragunath et Sivaranjani, 2015] proposed an ontology-based summarization approach that uses concept terms and feature vectors.
The system encodes the ontology using a tree structure, with each node representing a concept. The hierarchical classifier then selects sentences according to the tree-structured ontology to generate the summary. The system assigns a weight to each sentence based on its relevance to the concept represented by the node in the ontology.
To calculate the relevance of a sentence to a concept, the system uses a feature vector that represents the semantic content of the sentence. The feature vector includes information such as the frequency of concept terms and the presence of certain keywords.
The advantage of this approach is that it can generate a more accurate summary by considering the semantic relationships between the concepts in the text.

3.1.4 Lead and Body Phrase Method

[Tanaka *et al.*, 2009] proposed a summarization approach that searches for the same chunk in the lead and body sentences, which they call "triggers". Phrases are identified based on their similarity, and the body phrase is substituted into the lead phrase.
This process is done iteratively to generate new summary sentences. The system uses a scoring mechanism to select the best summary sentences based on the length of the sentence and its relevance to the topic.
One advantage of this approach is that it can capture the most important information in the text by identifying the triggers, which are typically the most important phrases or concepts in the text.

[Wasson, 1998] proposed a summarization method that uses searchable lead sentences to summarize news documents. The method involves converting each sentence of the document into a searchable query, and then selecting a set of lead sentences that match the most number of queries. The decision of sentence selection is based on Boolean retrieval, which involves

matching the query terms against the indexed document terms using Boolean operators such as AND, OR, and NOT. The selected lead sentences are then used to generate a summary.

3.1.5 Rule Based Method

[Genest et Lapalme, 2012] proposed a summarization approach based on an abstraction scheme. The goal is to condense the essential meaning of a text while minimizing repeated information in the summary. The approach involves identifying key noun and verb phrases using a scheme of events. Information extraction rules are applied to determine the most important noun and verb phrases associated with each event. A heuristic is then used to select the most suitable sentence for each event, considering factors like length, complexity, and relevance. The summary is generated using Simple Natural Language Generation (SNLG) rules to produce concise, grammatically correct sentences that convey the essential meaning. The aim is to create informative, easy-to-read summaries without redundancy.

[Kuppan et Sobha, 2009] proposed a summarization method that focuses on identifying coherent chunks of text within a document. They do this by applying a set of rules to the text and ranking it into well-organized chunks. To rank the chunks, they use a graph ranking algorithm that takes into account factors such as word frequency, word position, and string patterns to calculate the weight of each sentence. The algorithm then selects the most important sentences from each chunk to generate the summary.
The idea behind this approach is to identify the key topics and themes of the text by grouping together related sentences and analyzing their importance. By focusing on chunks rather than individual sentences, the method is able to capture the overall structure and meaning of the text more effectively. Additionally, the use of the graph ranking algorithm allows for a more nuanced approach to sentence selection, taking into account factors beyond simple word frequency. Overall, this approach represents a promising method for generating high-quality summaries that capture the essence of the original text.

3.2 Semantic Based Approach

3.2.1 Multimodal semantic model

[Greenbacker, 2011] proposed a summarization approach that works in three stages. In the first stage, an ontology is used to build a semantic model of the multimodal document. The ontology represents the relationships and connections between the concepts in the document. In the second stage, an information density matrix is used to rate each concept based on factors such as completeness of attributes and number of connections. This matrix assigns a score to each concept, which indicates its importance in the document. In the final stage, the summary is generated using the high-scoring concepts. The approach prioritizes concepts that are more complete and connected to other concepts, as these are likely to be more important for understanding the overall meaning of the document.

3.2.2 Information item based method

[Genest et Lapalme, 2011] proposed a framework for summarization that consists of four main stages: information item retrieval, sentence generation, sentence selection, and summary generation.

In the information item retrieval stage, the framework extracts SVO triplets (Subject-Verb-Object) from the input document. These triplets are used to represent the main concepts and relationships between them in the document.
In the sentence generation stage, the framework uses Simple Natural Language Generation (NLG) rules to generate sentences that express the extracted SVO triplets. This stage converts the extracted information items into human-readable sentences.
In the sentence selection stage, the framework ranks the generated sentences based on their document frequency. The most frequent sentences are considered more important and thus have higher chances of being selected for inclusion in the summary.
Finally, in the summary generation stage, the framework selects the top-ranked sentences to generate a summary. The selected sentences are organized in a coherent manner to create a summary that captures the main ideas and concepts of the input document.

3.2.3 Semantic Graph Based Method

[Moawad et Aref, 2012]proposed a summarization approach based on a rich semantic graph representation of the source document. The process involves creating a semantic graph from the document, where nodes represent concepts and edges represent relationships. Sentences are then ranked based on the weight of the words and sentences they contain. The highest-ranked sentence is selected, and a rich semantic graph is generated from it. Heuristic rules are applied to reduce the graph and produce an abstractive summary. The rich semantic graph approach captures the semantic meaning of the text, resulting in more accurate and informative summaries. However, it requires significant computational resources, and the heuristic rules may not always yield optimal results.

4 Conclusion

In conclusion, we have provided an overview of text summarization methods, with a distinction between extractive and abstractive methods. We have examined several different algorithms and approaches, including the TF-IDF method, cluster-based method, neural network-based method, fuzzy logic-based method, graph-based method, latent semantic analysis method, machine learning-based approach, and query-based summarization.

Furthermore, we have looked at abstractive approaches to text summarization, including structured-based and semantic-based approaches. We have highlighted the advantages and disadvantages of each method and their relevance for different types of texts and situations.

Chapter 3

Conception

1 Introduction

In this chapter, we provide a detailed explanation of the structure and methodology employed in our automated text document summarization system. We start by discussing the goals of text summarization, underscoring the significance of condensing essential information from extensive textual content.
Subsequently, we explore the global architecture of our system, emphasizing the key components and their respective roles in the summarization process. We carefully describe each stage of the summarization pipeline, clarifying the objectives and operations carried out at each step.

2 Objectif of our system

The goal of an automatic text summarization system is to generate a concise and coherent summary of a given text automatically, without human intervention. The system analyzes the input text, identifies important information, and produces a condensed version that captures the main ideas, key points, and relevant details. The objective is to assist users in quickly understanding the essence of the text, saving time and effort by eliminating the need to read through the entire document. Automatic text summarization systems can be useful in various applications, such as information retrieval, document organization, and assisting users in making informed decisions based on summarized content.[Hingu *et al.*, 2015]

3 Characteristics of article pages

In this thesis, our research revolved around the analysis and study of PDF documents. Within this section, we will delve into a comprehensive exploration of the various distinguishing characteristics of this pdf.see figure 3.1

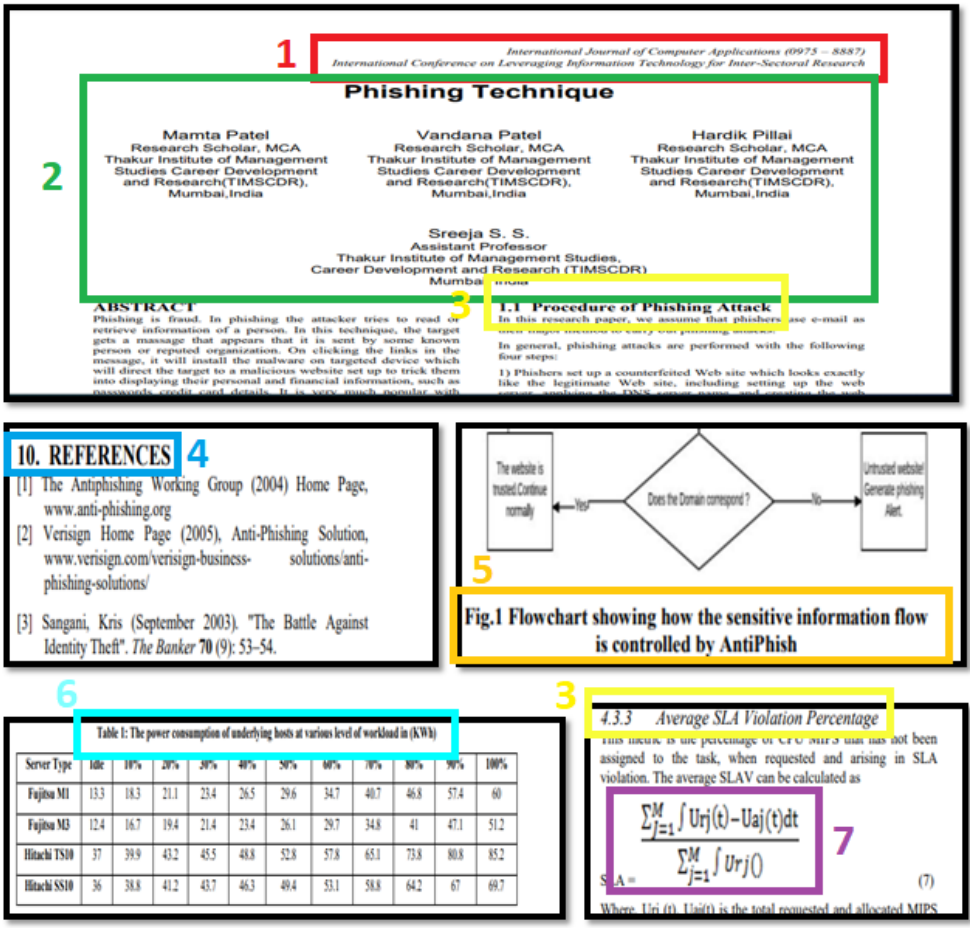

Figure 3.1: The fundamental components of the article pages that used in our approach

1: the presence of journal and conference titles introduces a level of complexity to text processing tasks. The diverse formatting and presentation styles adopted by different articles and conferences can present challenges when attempting automated analysis and summarization.

2: contain the header. The detection and display of headers in PDF documents are critical for document analysis and understanding. Headers contain important information such as document titles, section headings, and author names. Once the headers are detected and displayed, they can be removed from the text to create a concise summary and helps in focusing solely on the main content and reducing redundancy in the summary.

3: contains the titles of pdf. The extraction of titles enables efficient organization and categorization of PDF documents, allowing for improved document indexing, retrieval, and summarization.

4: contains the references of the pdf. To create a concise and focused summary, it is important to exclude the reference section from the document. The reference section typically contains a list of cited sources and is not directly relevant to the main content of the document. By omitting the reference section from the summary, the emphasis is placed on the core ideas and information presented within the document

5 and 6: contains teh titles of figures. To ensure a focused and concise summary, it is crucial to detect and remove both figure and table titles from the document. Figure titles, commonly found in separate caption sections or distinguished by specific formatting, provide detailed information pertaining to the figures or illustrations. Similarly, table titles convey essential information about the tables' content and organization.

7: contains equations. As can be seen here, the equations do not adhere to a standardized form or size. They vary from page to page and from one paper to another. This variability complicates the summarization process and can potentially mislead the approach by providing false information. That is why it is crucial to pay special attention to the equation components of the article, detect them accurately, and remove their content.

4 The architecture of our system

This architecture proposes a multi-step approach for performing automatic text summarization on PDF files. The process begins by opening the PDF and extracting its textual content. The extracted text then undergoes preprocessing to eliminate unwanted characters and standardize the format. Subsequently, the text is segmented into paragraphs to facilitate the representation of different sections within the document. TF-IDF weighting is then employed to calculate importance scores for words based on their frequency in the document and rarity in the corpus. Finally, cosine similarity is utilized to assess the similarity between paragraphs, enabling the selection of the most relevant ones for generating a summary.see figure 3.2

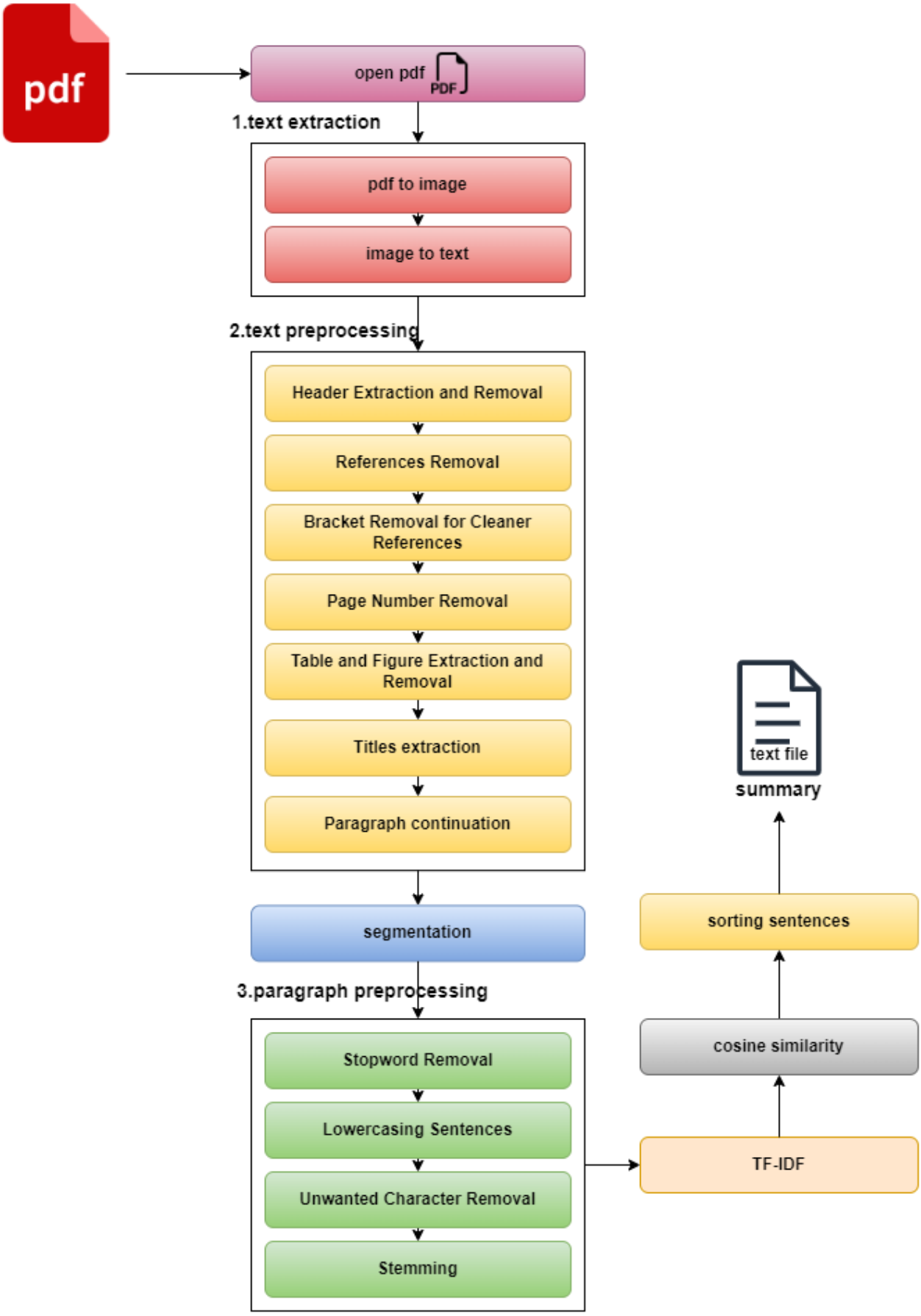

Figure 3.2: The architecture of the proposed text summarization system.

4.1 Text extraction

We have employed a specific technique in our approach to handle the commonly found two-column structure in articles. This complex structure poses challenges for traditional text extraction methods, resulting in inefficiency. Conventional techniques often fail to provide satisfactory results due to text overlap or merging across columns. To overcome this challenge, we employ the following technique: Firstly, we convert the PDF to images, and then we perform Optical Character Recognition (OCR) on each image to extract the text. Secondly, we combine the extracted text in the correct order, ensuring that the text from the first column is extracted before moving on to the second column. This preserves the original sequence of the content and enables more accurate and efficient text extraction from PDFs with a two-column structure. Figures 3.3 and 3.4 illustrate both the complex structure that can exist and the outcome of our technique. The detailed steps of our technique involve converting the PDF to images and then performing Optical Character Recognition (OCR) on those images to extract the text.

4.1.1 Pdf to images

In order to convert a PDF document into a series of images, we can follow this steps:

A. Input PDF: obtain the PDF File containing the content to be converted into images.

B. PDF Rendering: Utilize a PDF rendering engine to interpret the layout, fonts, and graphical elements of each page in the PDF.

C. Rasterization: Convert the vector-based content of the PDF into a rasterized format, transforming it into a grid of pixels that represent colors and shades.

D. Image Generation: Generate separate image files for each page of the PDF, preserving the visual appearance and layout of the original content.

E. Output Format: Choose an appropriate image file format, such as JPEG, PNG, or TIFF, for storing the generated images.

F. Image Manipulation: Perform additional operations on the converted images, such as resizing, cropping, rotation, or applying filters, as per specific requirements.

4.1.2 Image to text

For text extraction, we used conventional optical character recognition (OCR). It involves preprocessing the image, locating text regions, segmenting characters, extracting features, classifying characters, and performing post-processing.

A. Image Preprocessing: This step enhances the input image by resizing, scaling, reducing noise, and adjusting contrast to improve text legibility.

B. Text Localization: It analyzes the preprocessed image to identify regions that contain text, using techniques like edge detection, connected component analysis, and contour detection.

C. Character Segmentation: Once text regions are located, this step further analyzes each region to segment individual characters by splitting connected components within the text regions.

D. Feature Extraction: Features are extracted from each segmented character, such as stroke width, shape, and orientation. These features help differentiate between different characters during the recognition phase.

E. Classification: Machine learning algorithms are used to classify each segmented character based on the extracted features. A pre-trained model compares the features to recognize different characters, assigning a probability score to each character class.

F. Post-processing: After classification, post-processing techniques are applied to refine the recognized text. This includes spell-checking, language modeling, and contextual analysis to improve accuracy and coherence of the recognized text.

Figure 3.3: Example of article page with two columns

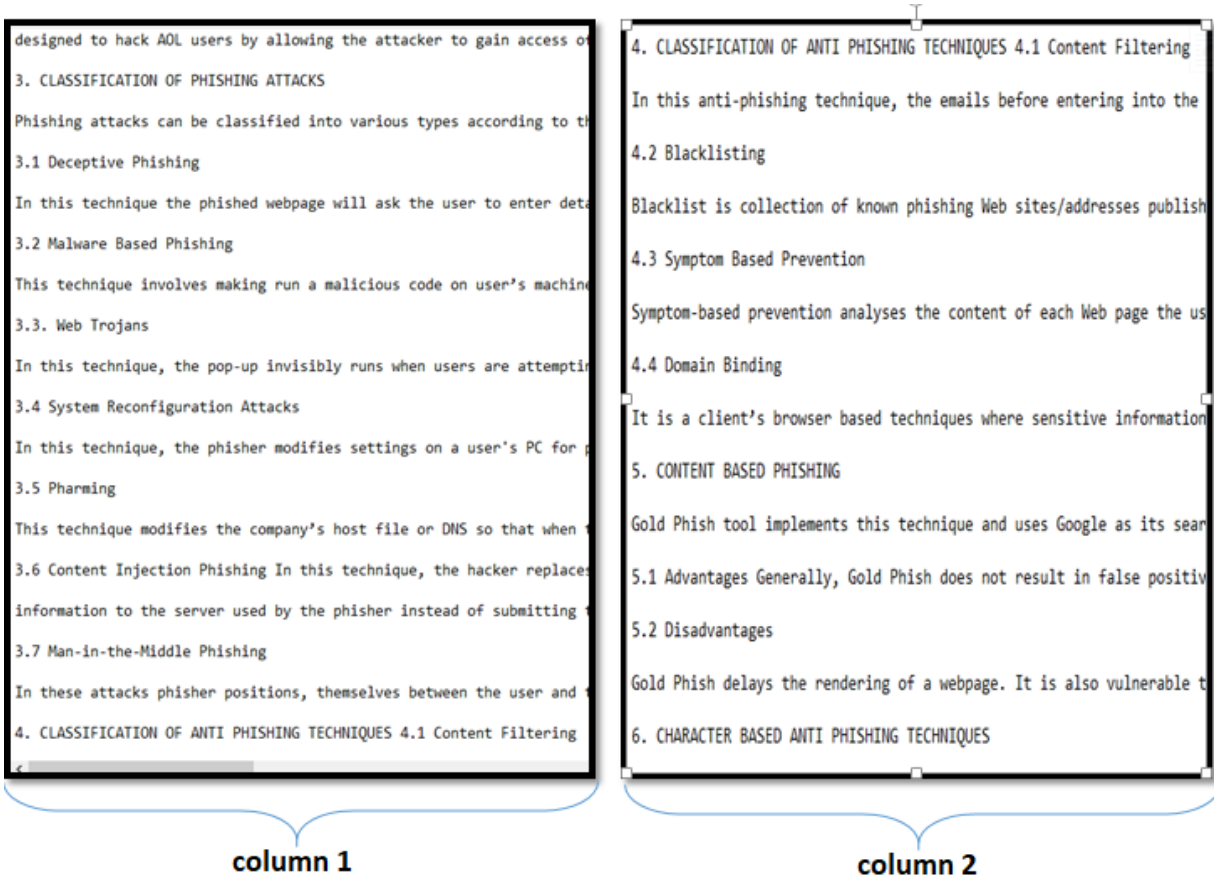

Figure 3.4: Example of the extracted text in two columns page

4.2 Document preprocessing

4.2.1 Enhancing Text Formatting in Two-Column article

In order to mitigate the formatting challenges arising from the two-column structure of the PDF documents, a post-processing approach was implemented to enhance the extracted text. After performing Optical Character Recognition (OCR) using the Tesseract OCR engine[Smith, 2007], the extracted text underwent a series of modifications. Firstly, line breaks within each paragraph were replaced with a space character to eliminate excessive line breaks. Secondly, a special character was introduced to indicate line continuation, replacing the hyphen and space commonly used in the original text. Finally, the special character was removed, effectively joining the lines that were previously separated by the hyphen and space. This post-processing methodology resulted in improved text formatting, reducing the excessive line breaks and spacing caused by the two-column layout. The modified paragraphs were then written to the output file, stored for further analysis, and displayed in the user interface for ease of reference.see figure 3.5

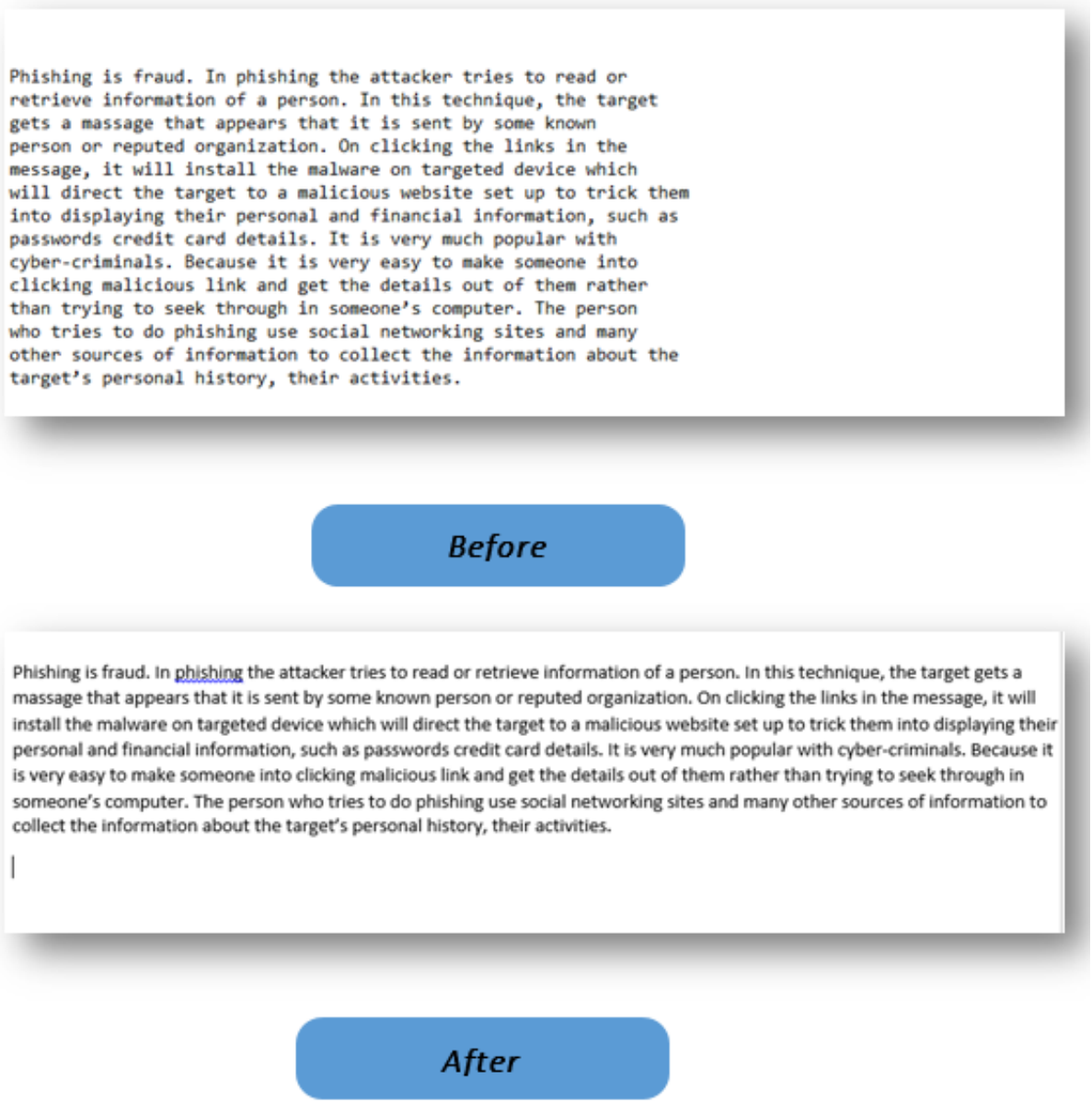

Figure 3.5: Example of Enhanced Text Formatting in Two-Column pages.

4.2.2 Header Extraction and Removal

After the text format enhancement, we extract the article heading from the first page using the following process: Firstly, we extract the header section and classify it as the article heading. Then, we remove the extracted header from the remaining text. This process exemplifies the enhanced document analysis achieved. By removing the header, the analysis becomes more focused and meaningful. See Figure 3.6

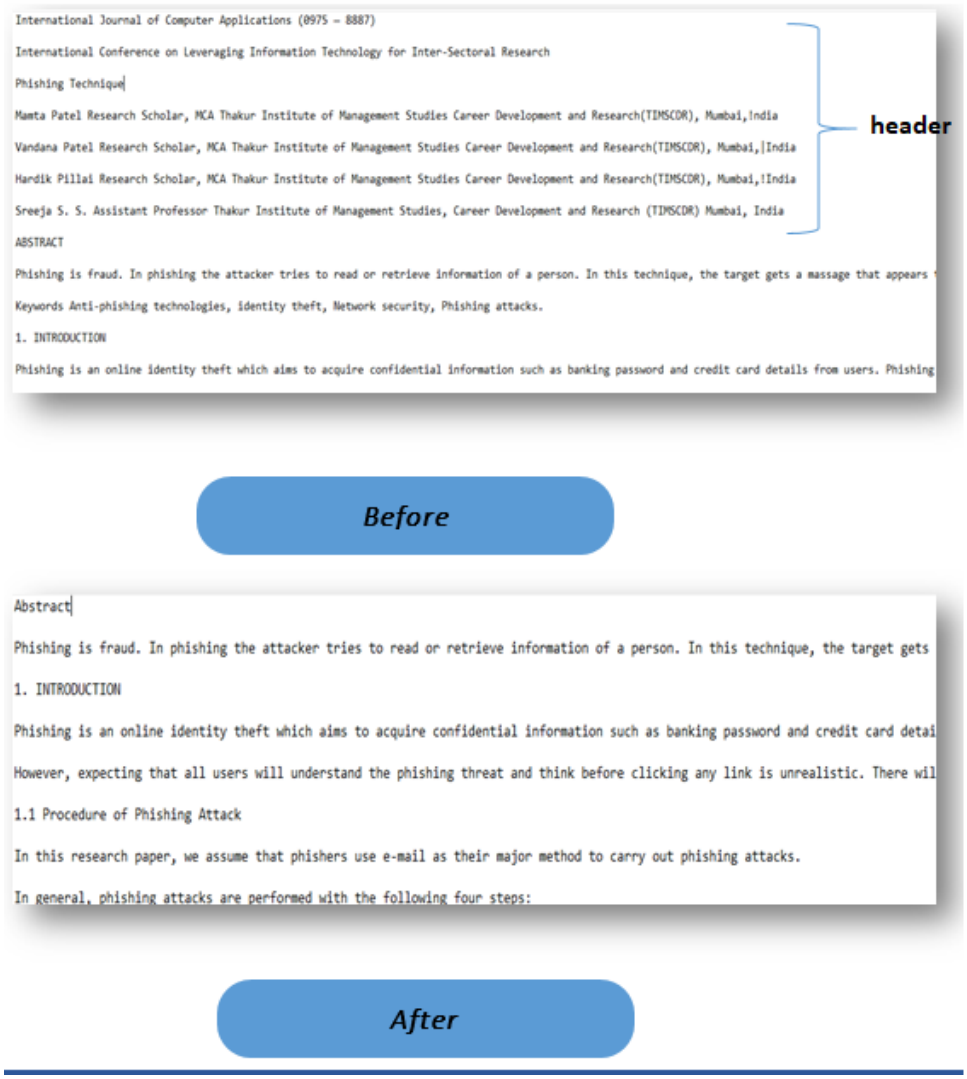

Figure 3.6: Example of article heading removal

4.2.3 References Removal

To make the input text of our approach more accurate, we remove the references section from the document. This step is undertaken to streamline the summarization process afterward by excluding the references and focusing solely on the main content of the document. By removing the references section, the subsequent analysis can be more targeted and specific to the relevant information within the document.

4.2.4 Bracket Removal for Cleaner References

The goal of this preprocessing step is to eliminate the square brackets used for referencing within the document. This step is commonly executed to enhance the readability and analysis of the text by removing the distracting reference markers, resulting in a cleaner and more coherent document.

4.2.5 Page Number Removal

By removing page numbers, the document becomes more streamlined and facilitates easier reading, resulting in a smoother and uninterrupted document analysis process.

4.2.6 Table and Figure Extraction and Removal

At this stage, we extract and eliminate tables and figures from the document. The main aim is to visually segregate these elements from the main textual content. By doing so, the summarization process will become more efficient and focused, as the main textual content can be analyzed independently without the presence of tables and figures. This approach enables a streamlined and targeted analysis, emphasizing the essential information within the text. see Figure 3.7

4.2.7 Titles extraction

The extraction and removal of article titles are also crucial steps in ensuring clarity and eliminating any misleading information from the text. As depicted in Figure ??, all the article titles are successfully extracted, demonstrating that each section can be identified independently. This process enhances the readability and organization of the text, allowing for a better understanding of the article's structure and content.see figure 3.7

4.2.8 Paragraph continuation

The final step in this stage is dedicated to addressing the challenge of paragraphs that extend over multiple pages in a document. Its primary objective is to enable a smooth interpretation of these multipage paragraphs, promoting coherence and facilitating a comprehensive understanding of the text during the document analysis process. The effectiveness of the PDF preprocessing techniques utilized in this study is exemplified by a compelling result, depicted in Figure ??. This result highlights the successful resolution of multipage paragraphs, underscoring the enhanced interpretability of the text.see figure 3.7

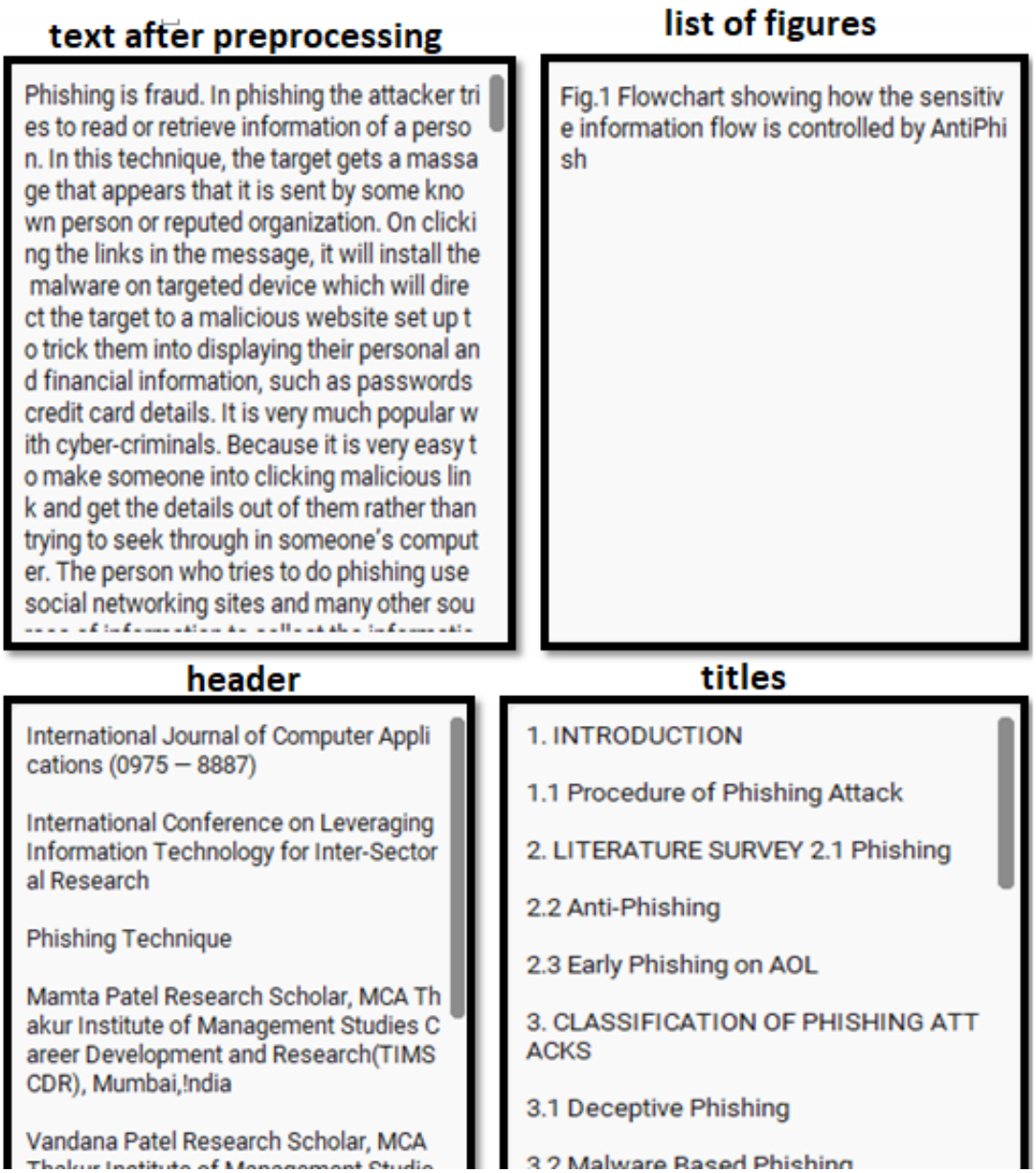

Figure 3.7: Example of elements extracted in the preprocessing phase

4.3 Segmentation

After successfully completing the preprocessing document phase, we proceed to segment the document text into paragraphs and further into individual words. This crucial segmentation step plays a pivotal role in text summarization, leading to enhanced readability, improved organization, and ultimately enabling a more precise interpretation of the document's content. Figure 3.8 visually illustrates the segmentation phase, depicting the transformation from the overall content text to its constituent words.figure3.8

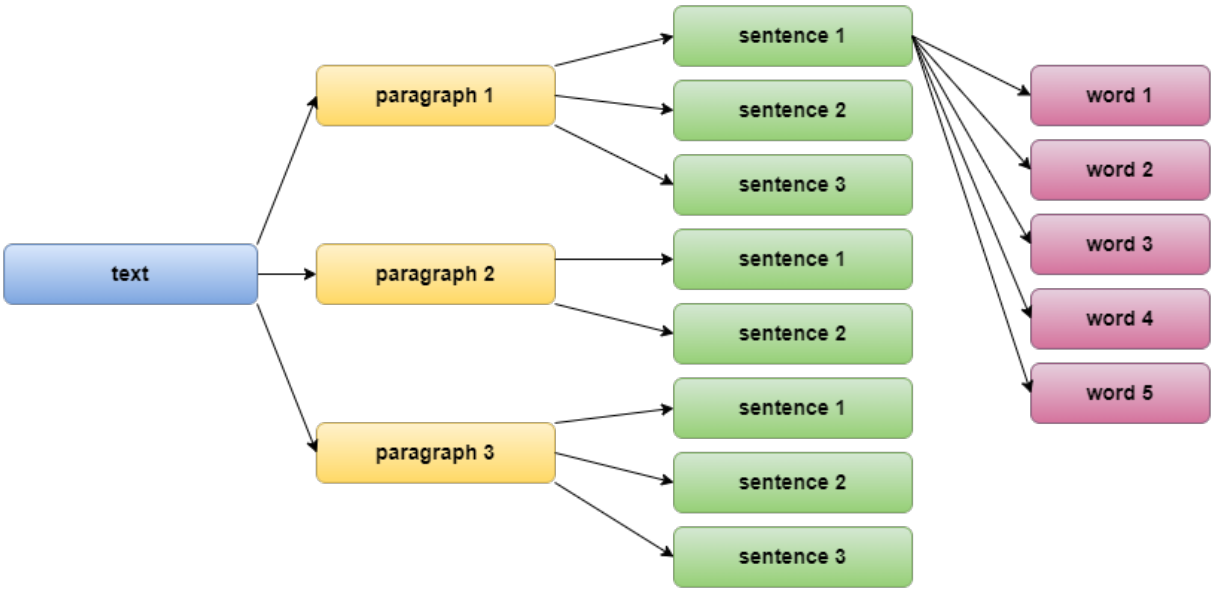

Figure 3.8: segmentation architecture

4.4 Paragraphs preprocessing

4.4.1 Stopword Removal

This phase encompasses all the essential steps for text preprocessing prior to applying summarization methods. These steps include:
By eliminating frequently occurring words that possess limited semantic significance, the subsequent analysis process becomes more targeted towards pertinent and substantial content. This omission of stopwords contributes to an improved level of accuracy and precision in document analysis, facilitating a more profound comprehension of the textual information at hand.

4.4.2 Lowercasing Sentences

This step ensures a consistent and uniform text case, thereby enhancing the process of text processing and analysis. Lowercasing sentences eliminates the concern of case sensitivity, enabling more accurate text matching, extraction, and interpretation.

4.4.3 Unwanted Character Removal

After removing stopwords and transforming sentences into lowercase, the next step is to eliminate all undesirable characters, such as punctuation marks and quotation marks, from the document. This removal process ensures a cleaner and more comprehensible text, enhancing readability and minimizing potential disruptions during subsequent analysis tasks.

4.4.4 Stemming

Stemming is the process of reducing words to their base or root form, which is accomplished using stemming algorithms. Implementing stemming optimizes text representation by reducing word variations, resulting in improved efficiency and accuracy in subsequent analysis tasks. Stemming plays a crucial role in enhancing various natural language processing tasks, including

text similarity calculations and topic modeling, as it helps group related words together, thereby facilitating more effective analysis and interpretation. See Figure 3.9

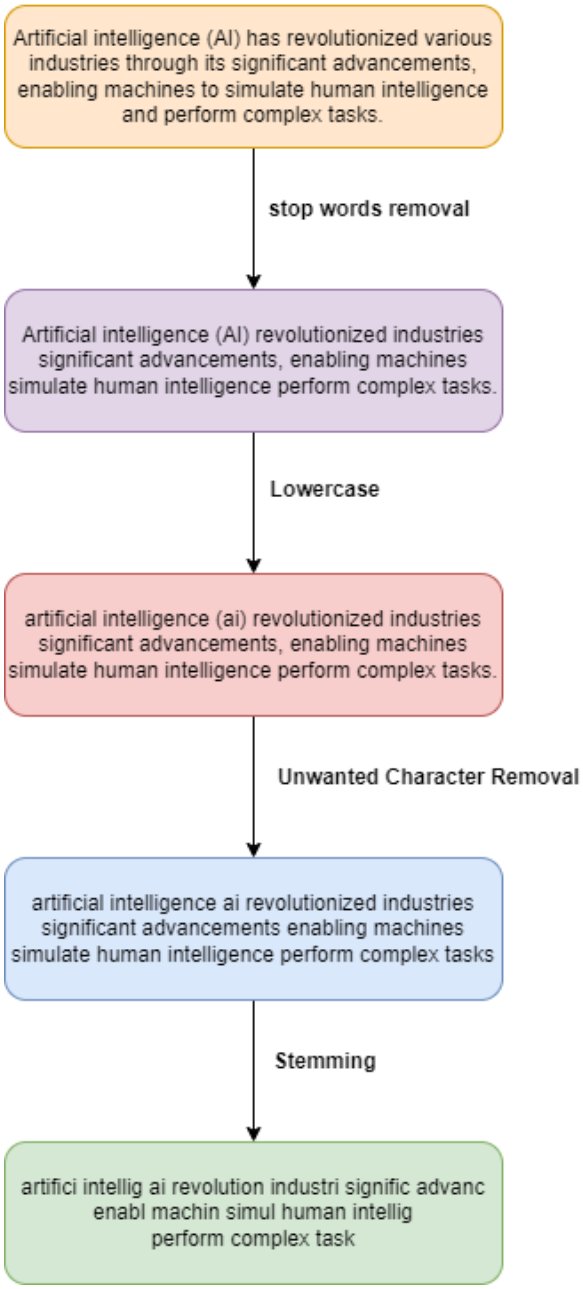

Figure 3.9: Example of applying the text preprocessing steps.

4.5 Term Frequency-Inverse Document Frequency

TF-IDF (Term Frequency-Inverse Document Frequency) [Luhn, 1957]is a statistical measure used to evaluate the importance of a term in a document within a collection or corpus of documents. It consists of two components: term frequency (TF) and inverse document frequency (IDF).

4.5.1 Term Frequency (TF)

The term frequency measures the frequency of a term (word) within a specific document. It indicates how often a term appears in a document relative to the total number of terms in that document. TF can be calculated using different methods, such as raw term frequency (count of occurrences) see equation 3.1 or using a logarithmic scale to attenuate the effect of very frequent terms.

$$\mathrm{TF}(t,d) = \frac{\mathrm{count}(t,d)}{|d|} \tag{3.1}$$

TF(t,d): This represents the term frequency of a term **t** in document **d**. It measures the frequency of the term **t** within the document **d**. Essentially, it calculates how often the term appears in the document.

count(t,d): This represents the count of occurrences of the term **t** in the document **d**. It is the actual number of times the term appears in the document.

$|d|$**:** This represents the length or the total number of terms in the document **d**. It measures the size or the length of the document, typically represented by the total number of words or terms it contains.

4.5.2 Inverse Document Frequency (IDF)

The inverse document frequency is a measure of the rarity or uniqueness of a term across all documents in the corpus. It helps to identify terms that are more informative or distinctive. IDF is calculated by dividing the total number of documents in the corpus by the number of documents that contain the term. The resulting value is then logarithmically scaled to dampen the effect of highly frequent terms.see equation 3.2

$$\mathrm{IDF}(t) = \log\left(\frac{N}{\mathrm{DF}(t)}\right) \tag{3.2}$$

IDF(t): This represents the inverse document frequency (IDF) of a term **t**.

log: This is the logarithm function.

N: This represents the total number of documents in the collection. It is the denominator in the ratio and provides the overall context of the document corpus.

DF(t): This represents the document frequency of the term **t**. It measures how many documents in the collection contain the term **t**. A higher document frequency indicates that the term is more common or less informative, whereas a lower document frequency suggests that the term is more rare or more informative.

to transform it into a logarithmic scale. The logarithm function helps in reducing the impact of very high document frequencies. The TF-IDF value of a term in a document is obtained by multiplying its TF by its IDF. The higher the TF-IDF score of a term in a document, the more important or relevant that term is to that document compared to the rest of the corpus. Terms that appear frequently in a particular document but rarely in other documents tend to have higher TF-IDF scores. see equation 3.3

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \text{IDF}(t) \tag{3.3}$$

TF-IDF(t,d): This represents the term frequency-inverse document frequency (TF-IDF) of a term **t** in a document **d**. It is a commonly used weighting scheme in information retrieval and text mining to evaluate the importance of a term within a specific document in the context of a document collection.

TF(t,d): This term represents the term frequency of a term **t** in a document **d**. It measures the frequency of the term **t** within the document **d**. Essentially, it calculates how often the term appears in the document.

IDF(t): This term represents the inverse document frequency (IDF) of a term **t**. It is a measure used to assess the importance or rarity of a term within a collection of documents. It calculates the logarithm of the inverse ratio of the total number of documents to the document frequency of the term **t**.

4.6 Cosine similarity

By calculating the cosine similarity between two sentence vectors, it is possible to assess their similarity in terms of content and meaning. This approach enables the identification of closely related sentences and can be leveraged in various text analysis tasks such as document clustering, sentence similarity ranking, or summarization. Higher cosine similarity scores between sentence pairs indicate greater similarity in terms of word usage and contextual meaning, while lower scores suggest less similarity or dissimilarity. By applying cosine similarity across all sentence pairs in a document, it becomes possible to gain insights into the semantic relationships and connections between different parts of the text.

$$\text{cosine\_similarity}(a,b) = \frac{a \cdot b}{\|a\|\|b\|}$$

cosine_similarity(a,b)**:** This represents the cosine similarity between two vectors **a** and **b**. Cosine similarity is a measure used to determine the similarity between two vectors based on the cosine of the angle between them.

$\mathbf{a} \cdot \mathbf{b}$**:** This term denotes the dot product of vectors **a** and **b**. The dot product is a mathematical operation that calculates the sum of the products of corresponding elements in the vectors.

$\|\mathbf{a}\|$ **and** $\|\mathbf{b}\|$**:** These terms represent the norms (or magnitudes) of vectors **a** and **b** respectively. The norm of a vector is a measure of its length or magnitude in a vector space. In the equation, the norms of **a** and **b** are multiplied together.

4.7 Sorting sentences

The sorting operation arranges the sentences in descending order of their cumulative scores, ensuring that the most important sentences appear at the top. This allows for the straightforward selection of the top-ranked sentences to construct a concise summary that captures the essential information from the original text.

4.8 Summary generation

In the final step of the system, summary generation takes place. This crucial stage aims to distill the essential information from the input text into a concise and coherent summary.see figure3.10

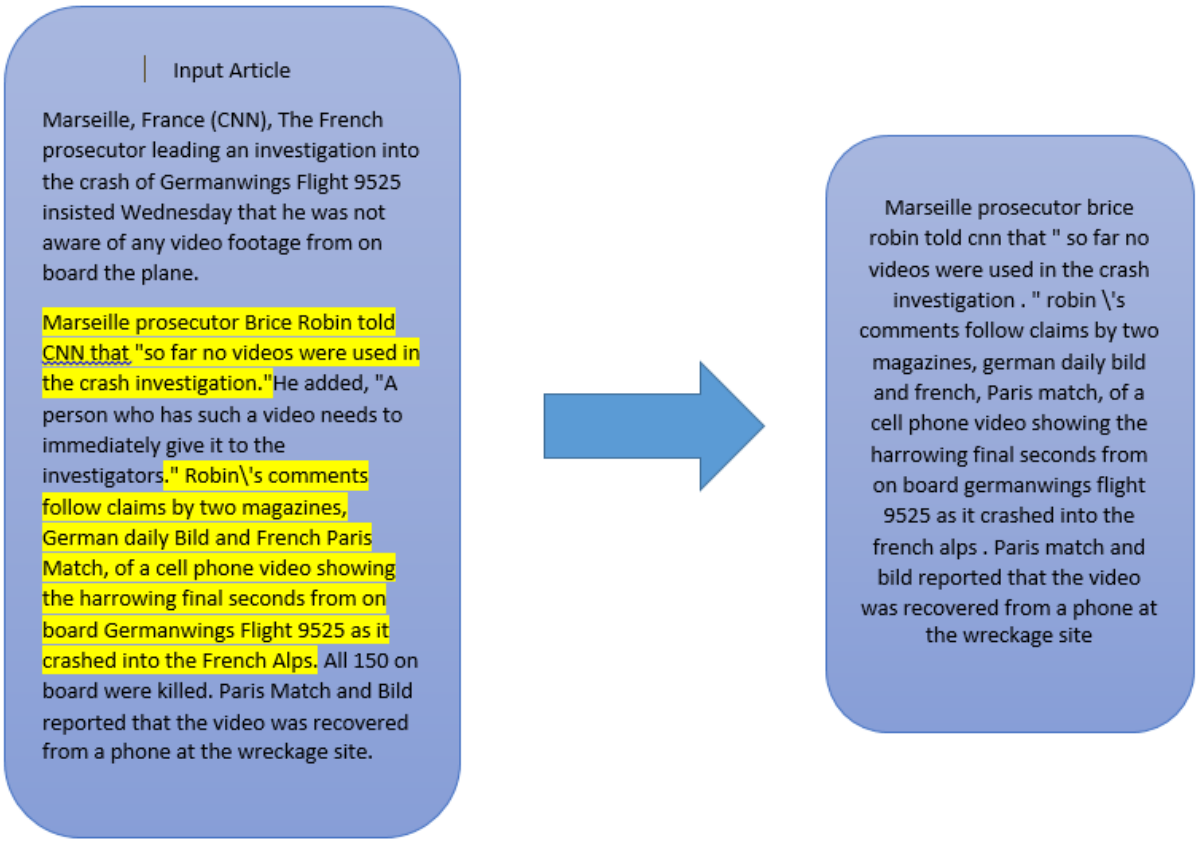

Figure 3.10: summary exemple

5 Conclusion

To summarize, this chapter provides a detailed explanation of the structure and methodology employed in an automated text document summarization system. It emphasizes the importance of condensing essential information from extensive textual content and explores the global architecture of the system. Each stage of the summarization pipeline is carefully described, clarifying the objectives and operations carried out at each step.

Chapter 4

Implementation

1 Introduction

In this chapter, we provide an overview of the system's implementation. Firstly, we outline the development tools employed for its construction. Subsequently, we present a comprehensive examination of the interfaces that are offered by the system.

2 Environment

The implementation and testing of our application were conducted in the following hardware and software environment:

Table 4.1: System Specifications

Model Part	**Used Laptop**
Installed memory (RAM)	4.00 Go
Processor	Intel(R) Core(TM) i5-3317U CPU @ 1.70GHz 1.70 GHz
Operating System	Windows 10 Professionnel
System Type	Système d'exploitation 64 bits, processeur x64

3 Software environment

3.1 Python

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python emphasizes code readability and provides a clean syntax that allows programmers to express concepts in fewer lines of code compared to other languages. It has a large standard library and a thriving ecosystem of third-party libraries and frameworks, making it suitable for various applications such as web development, data analysis, machine learning, scientific computing, automation, and more. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming, allowing developers to choose the

approach that best fits their project requirements. Due to its ease of use and extensive community support, Python has gained popularity among beginners and experienced developers alike.[Van Rossum et Drake Jr, 1995]

3.2 Pycharm iDE

PyCharm is an integrated development environment (IDE) specifically designed for Python programming. Developed by JetBrains, PyCharm provides a comprehensive set of tools and features that facilitate efficient coding, debugging, and testing of Python applications. It offers a user-friendly interface, advanced code editor with syntax highlighting, code completion, and error detection, as well as powerful refactoring capabilities to improve code quality and maintainability.[Islam, 2015]

4 libraries

4.1 Math

The math library in Python is a standard library module that provides various mathematical functions and operations. It includes a wide range of functions for performing mathematical calculations, working with numbers, and manipulating numerical data. The math module is part of the Python Standard Library, which means it is included with every standard installation of Python, and you can readily access its functionality without needing to install any additional packages.

4.2 Numpy

NumPy is a Python library for numerical computing. It allows you to work with large arrays of numbers efficiently and provides a collection of mathematical functions. It is widely used in scientific and data analysis applications due to its speed and versatility. NumPy simplifies tasks like creating arrays, performing mathematical operations, and manipulating data. It is an essential tool for numerical computations in Python.[Oliphant *et al.*, 2006]

4.3 NTLK

The NLTK (Natural Language Toolkit) library is a popular open-source Python library for working with human language data. It provides a wide range of tools and resources for tasks such as text processing, tokenization, stemming, part-of-speech tagging, parsing, semantic reasoning, and more. NLTK is widely used in the fields of natural language processing (NLP) and computational linguistics[Hardeniya *et al.*, 2016]

4.4 Costumetkinter

CustomTkinter is a Python library that extends the functionality of Tkinter, a popular GUI toolkit. It provides additional features and widgets, making it simpler to create sophisticated and robust Python user interfaces. With CustomTkinter, you have access to a wide range of pre-built widgets and functions that facilitate the construction of custom GUIs. The library offers tools for developing tailored user interfaces, including custom widgets, dialogues, and menus.[Seetha *et al.*, 2023]

4.5 Re

The "re" library in Python is a standard library module that provides support for regular expressions (regex). Regular expressions are patterns used to match and manipulate strings. The "re" library allows you to work with these patterns and perform various operations such as pattern matching, searching, substitution, and splitting of text.[Chapman et Stolee, 2016]

4.6 Pdf2image

The "pdf2image" library in Python is a popular third-party library that allows you to convert PDF documents into a series of images. It provides a convenient way to extract individual pages or all pages from a PDF file and save them as image files, such as JPEG, PNG, or TIFF.[Harmaakivi, 2022]

4.7 Os

The "os" library in Python is a standard library module that provides a way to interact with the operating system. It offers functions and methods to perform various operating system-related tasks, such as file and directory operations, environment variables, process management, and more.

4.8 Natsort

The "natsort" library is a Python package that provides natural sorting functionality for lists and other iterable objects. "Natural sorting" refers to a sorting algorithm that orders strings in a way that is more human-friendly and intuitive compared to standard lexicographic sorting.

4.9 PIL

PIL is a library for handling and manipulating images in Python. It provides functionality for opening, manipulating, and saving images in various formats, similar to Pillow. However, as of my knowledge cutoff in September 2021, the original PIL library is no longer actively maintained, and Pillow has become the more commonly used and actively developed library for image processing in Python. [Umesh, 2012]

4.10 Pytesseract

The "pytesseract" library is a Python wrapper for the Tesseract OCR (Optical Character Recognition) engine. OCR is a technology that enables the extraction of text from images or scanned documents, allowing the computer to interpret and process the text data.

Tesseract is an open-source OCR engine developed by Google. It has the capability to recognize and extract text from various image formats, including scanned documents, photographs, screenshots, and more. The pytesseract library provides a convenient way to use Tesseract OCR in Python. [Sharma *et al.*, 2022]

5 Corpus

For this study, a corpus of conference articles was collected from ijcaonline . The website hosts a vast collection of conference proceedings from various categories, covering a wide range of

research domains. The corpus consisted of a diverse set of conference articles, totaling approximately 11 pages of content.
The articles were provided in PDF format, presenting a variety of structured data, including textual content, tables, graphs, figures, equations, and other data representations. The corpus encompassed articles from different conferences, each with its own unique structure and formatting conventions.
This corpus served as a valuable resource for evaluating and applying our proposed approach for summarization. It provided a representative sample of conference articles, enabling us to study the effectiveness of our approach across different domains and content types.
It's important to note that the corpus was carefully selected and extracted from ijcaonlinebased on specific criteria related to relevance and availability. The articles were processed and preprocessed to extract the necessary textual content for the summarization task.
Throughout this dissertation, we refer to this collection of conference articles from ijcaonline as the "Conference Article Corpus." The corpus played a significant role in the development, evaluation, and analysis of our proposed summarization approach.

6 System overview

The main modules of our application are summarized in the following figure.4.1

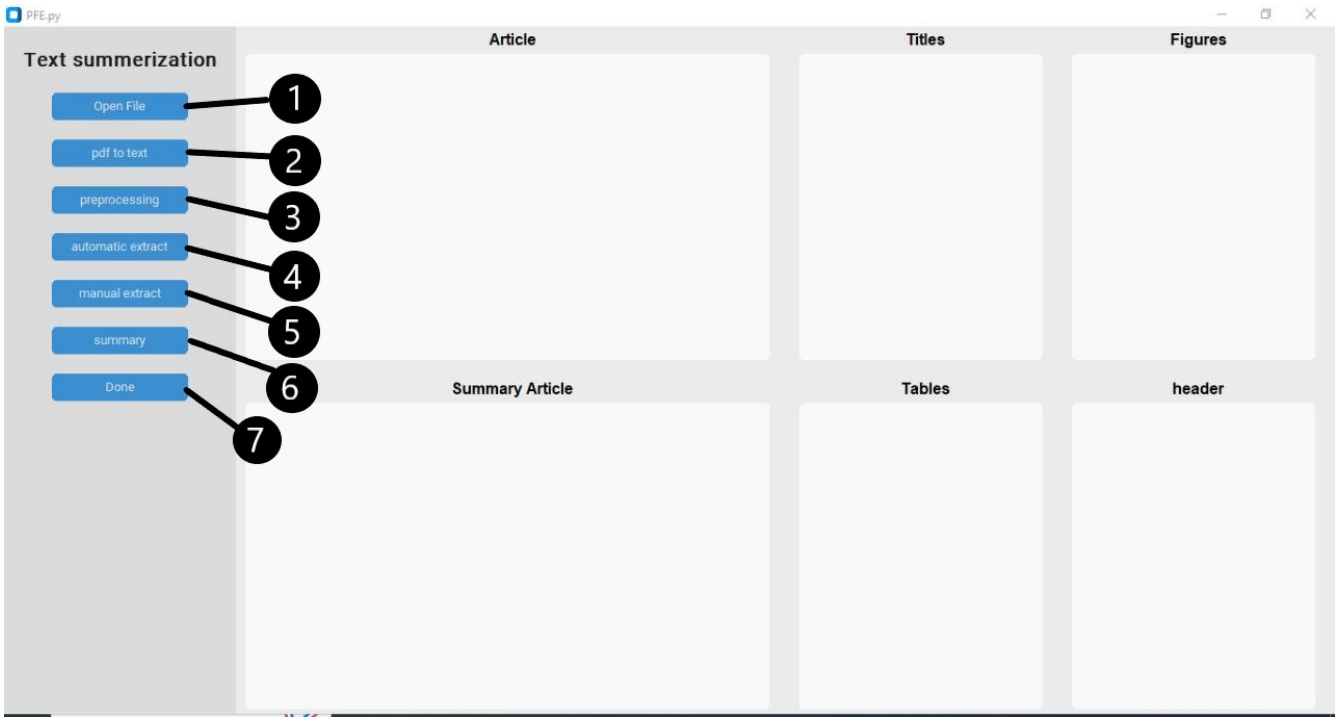

Figure 4.1: Interface presentation

1 : This button opens the PDF file.

2 : This button converts the PDF pages to text.

3 : This button applies preprocessing to the extracted text from the PDF file.

4 : This button is responsible for the automatic selection ofarticle sentences.

5 : This button is responsible for the manual selection of article sentences.

6 : This button generates and displays the summary.

7 : This last button finishes the summarization process.

7 Usage scenario

To provide a concrete illustration of the practical application of our theoretical proposal, we present a usage scenario that will highlight the advantages and functionalities of our implementation.

First, we click on the "Open PDF" button to open the PDF. This action enables us to choose the specific document that we wish to summarize.

Next, by clicking the "PDF to Text" button, we employ various methods to convert the PDF into text. Firstly, we utilize the pdf2image library to convert the PDF pages into a sequence of images. Then, we process these images using optical character recognition (OCR) techniques, powered by the Tesseract OCR engine, to extract the text from them. The figure below displays the results of the phase.see figure 4.2

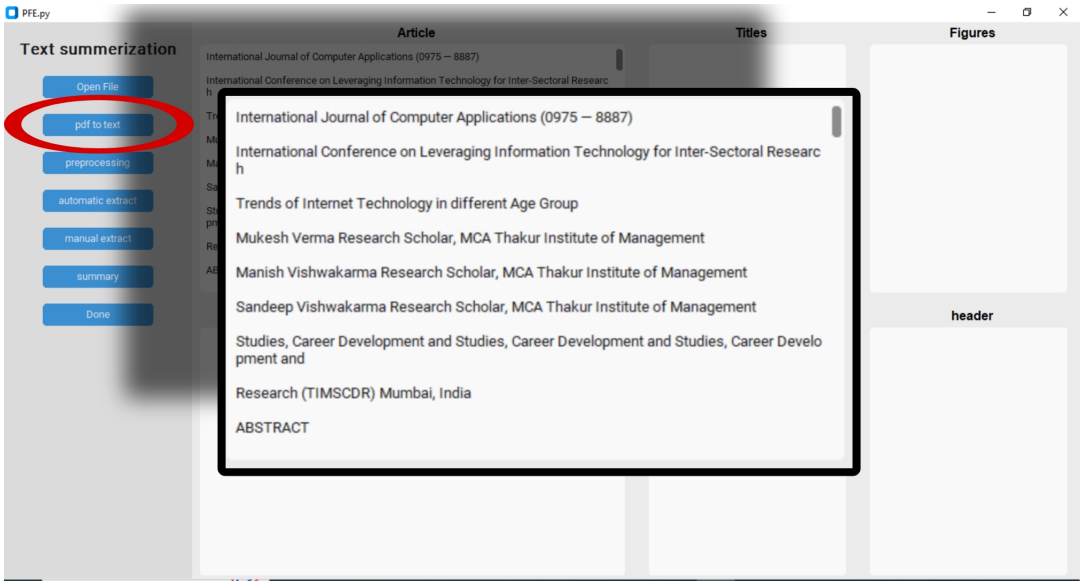

Figure 4.2: Extracted text

After extracting the text, we apply various preprocessing methods to enhance its quality and achieve a more refined representation,Once the preprocessing methods are applied, the text is organized as shown in the figures below: (Figure 4.3, Figure 4.4, Figure 4.5,Figure 4.6, and Figure 4.8). Each figure illustrates the extraction of a specific label, encompassing the title sections, figure legends, table captions, and article headings and subheadings.

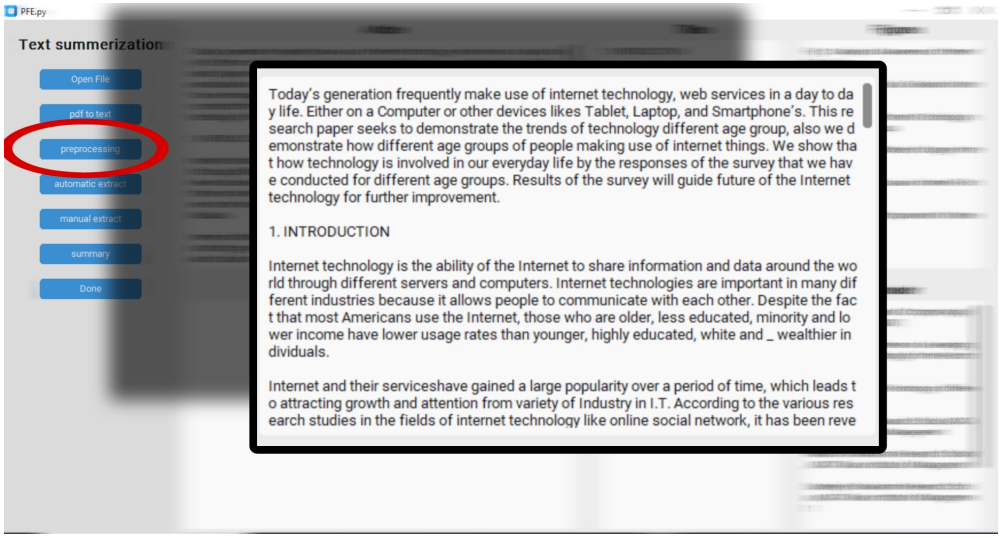

Figure 4.3: An example of the preprocessed document text.

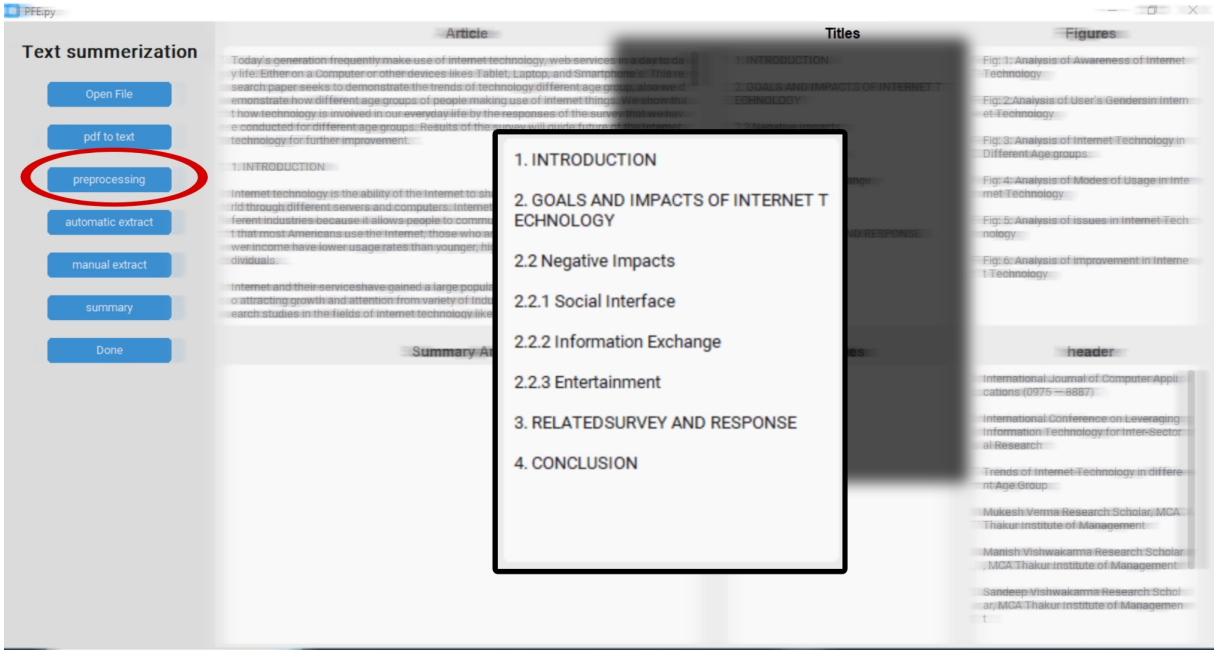

Figure 4.4: The extraction of main titles within the article.

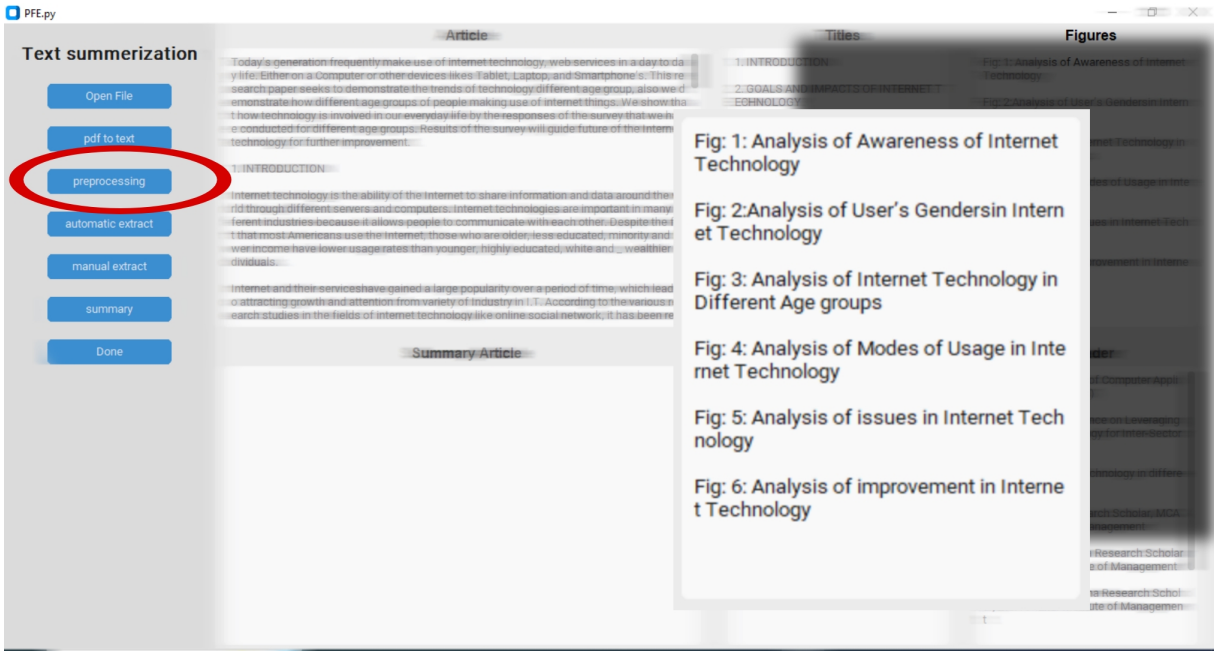

Figure 4.5: The extraction of figure legends within the article.

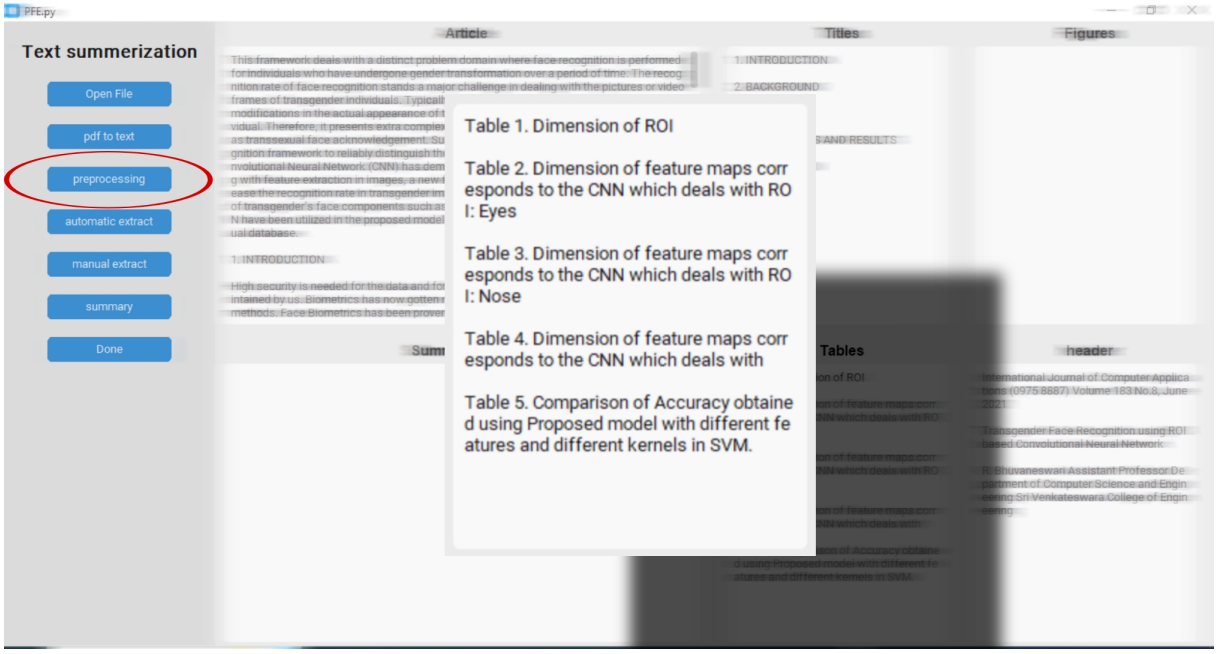

Figure 4.6: The extraction of table captions within the article.

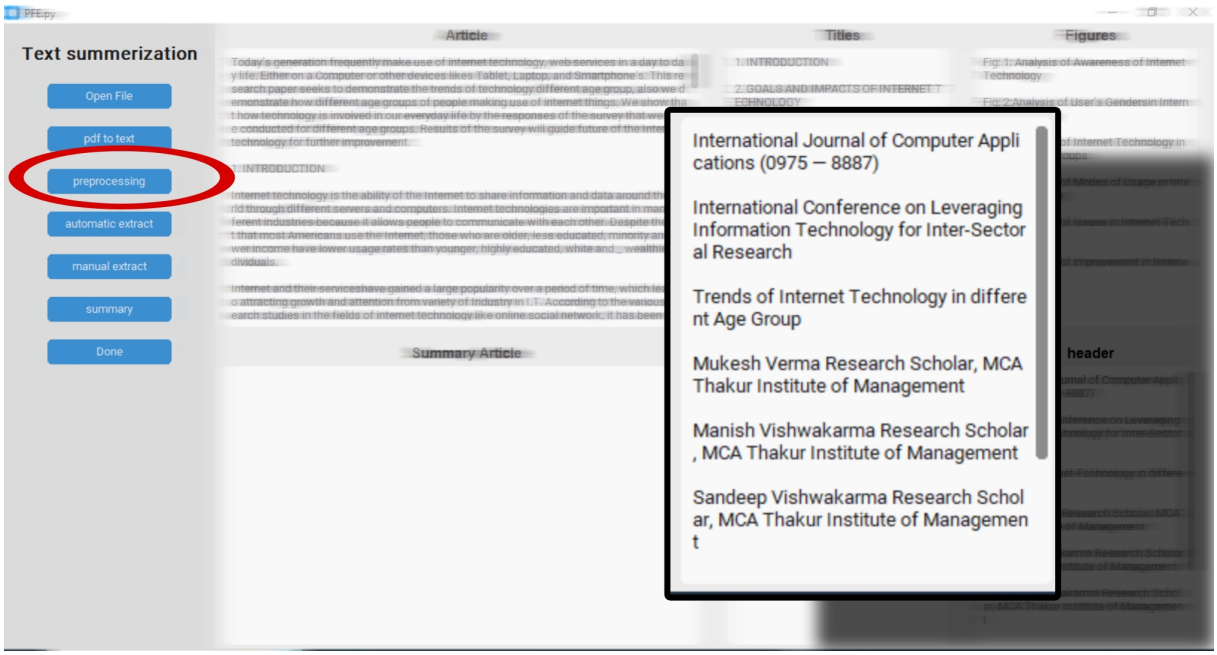

Figure 4.7: The extraction of article heading and sub-heading condition.

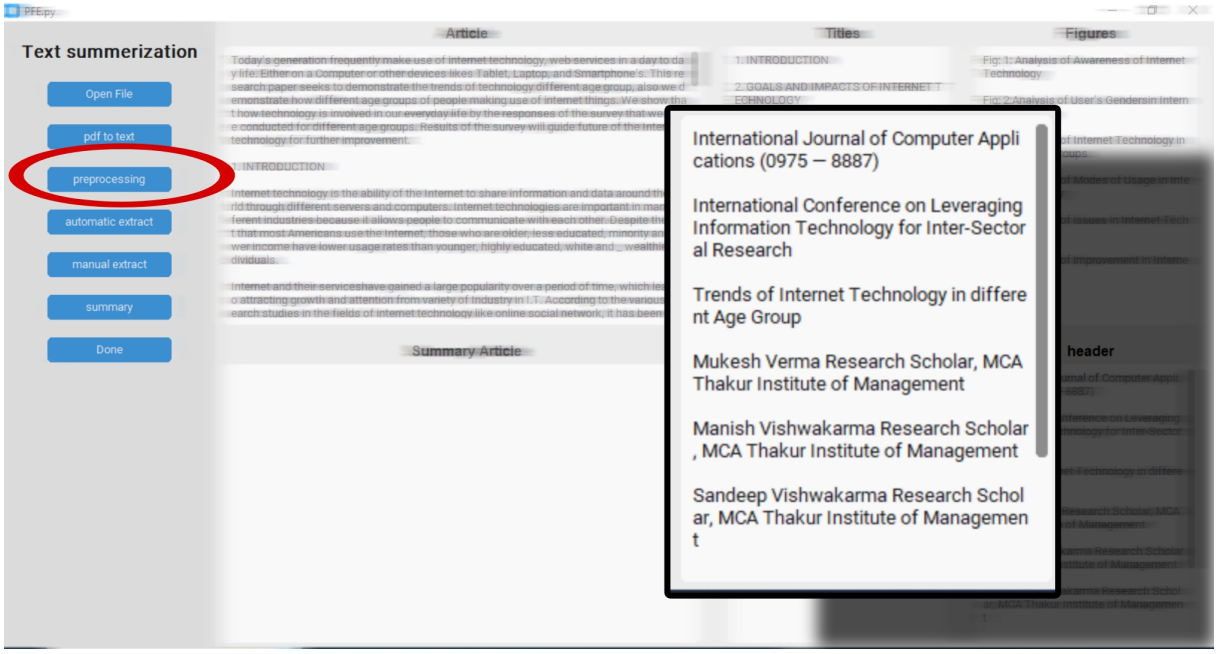

Figure 4.8: The extraction of article heading and sub-heading.

Now, we have two types of sentence selection methods to extract the important sentences: automatic selection and manual selection.

- In the automatic selection method, the number of sentences to be selected is automatically determined through various techniques and algorithms.see figure 4.9

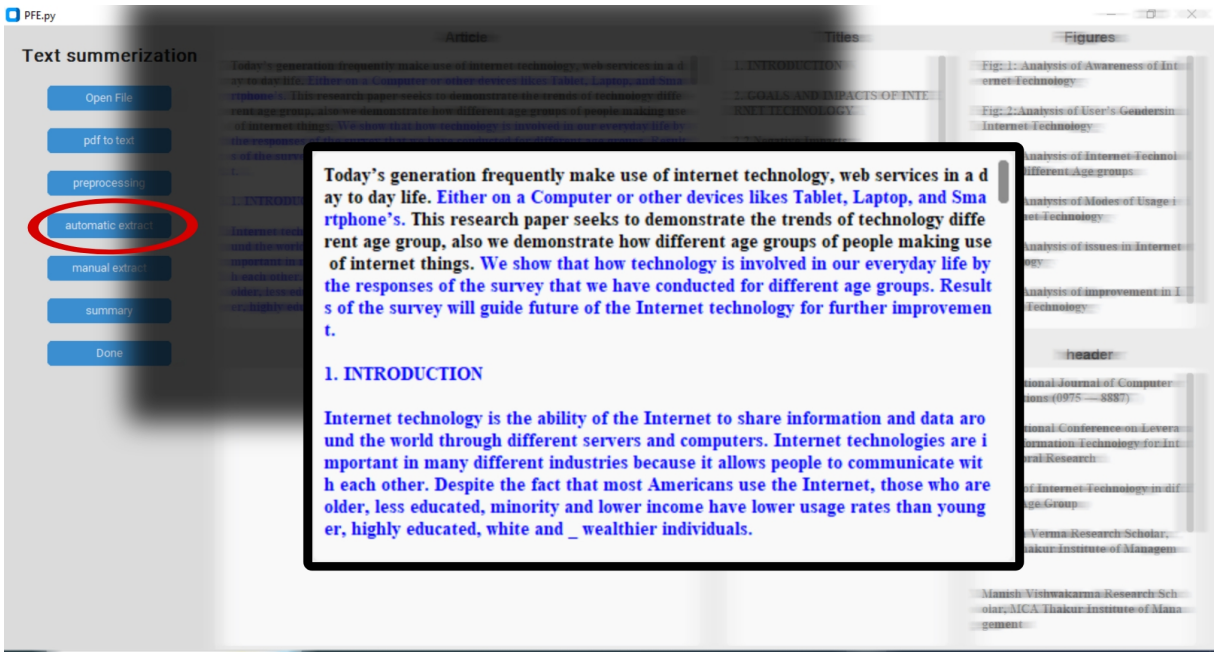

Figure 4.9: Automatic selection

- On the other hand, in the manual selection method, the number of sentences to be selected is determined by users through a manual process.
  The figure below displays the selection result using the manual selection method.see figure 4.10

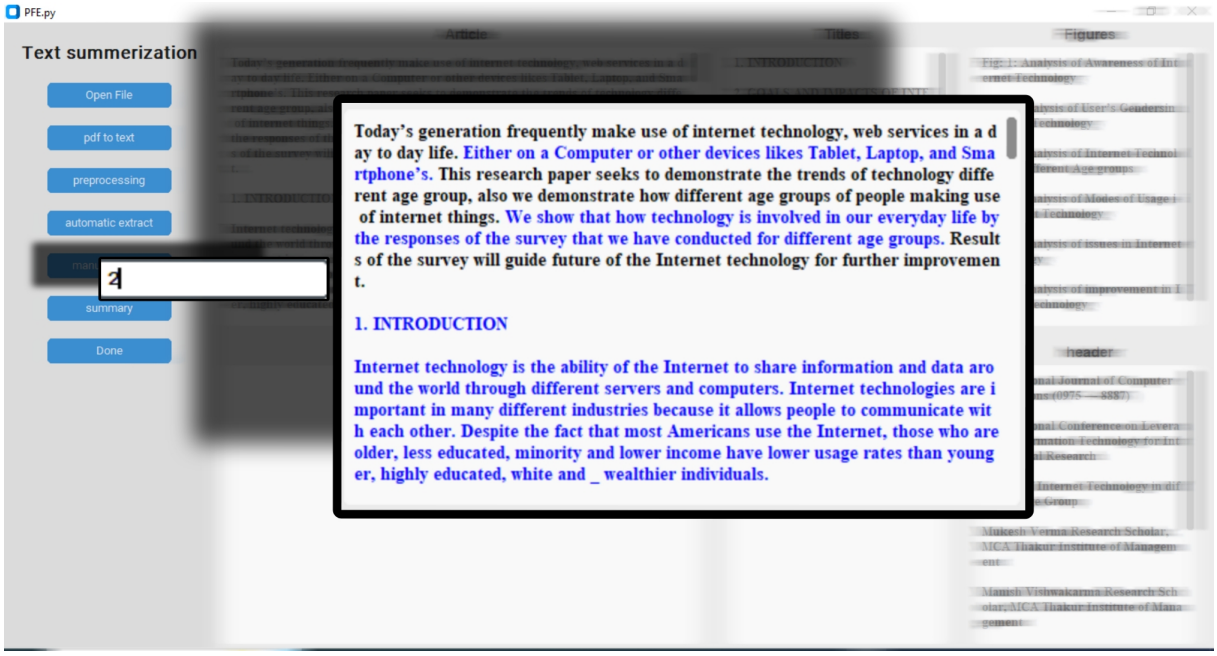

Figure 4.10: Manual selection

In order to select important terms for the text summarization process, we employed the TF-IDF (Term Frequency-Inverse Document Frequency) technique.see figure 4.11

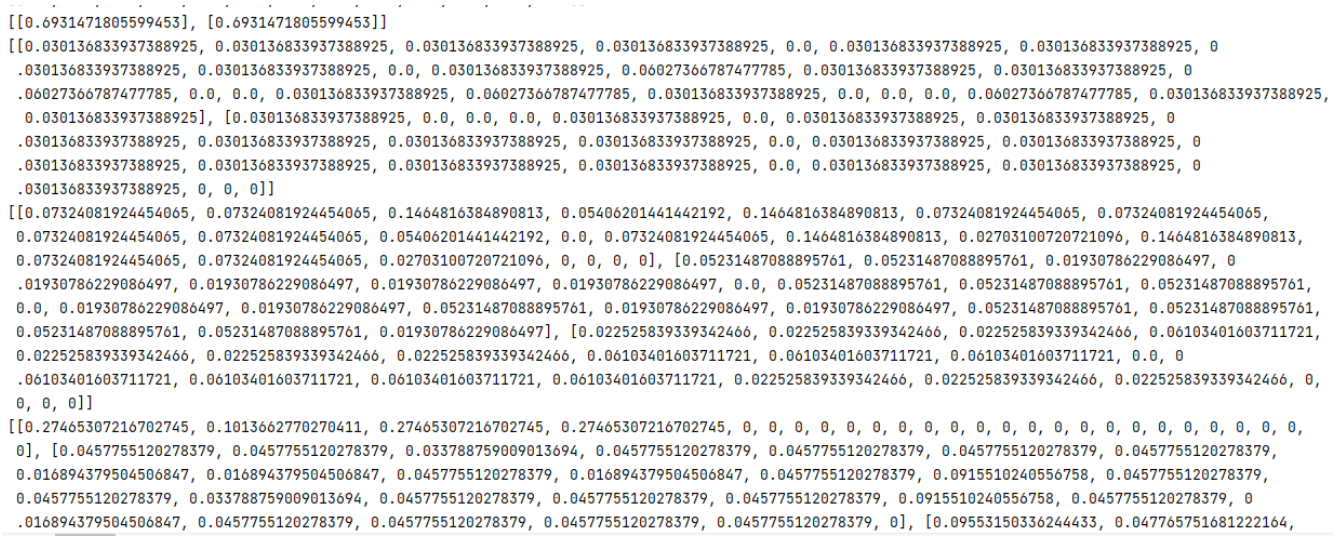

Figure 4.11: TF-IDF results

To complement the TF-IDF approach, we also incorporated cosine similarity methods to enhance the text summarization process. After applying TF-IDF to compute the term importance scores, we utilized cosine similarity to measure the similarity between sentences or document representations.see figure 4.12

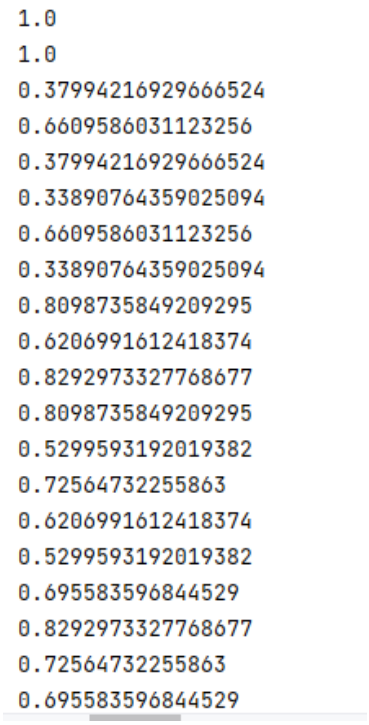

Figure 4.12: Cosine similarity results

The final step is the summary generation, where the extracted important sentences are utilized to generate a concise summary. This summary is then displayed in the user interface.see figure 4.13

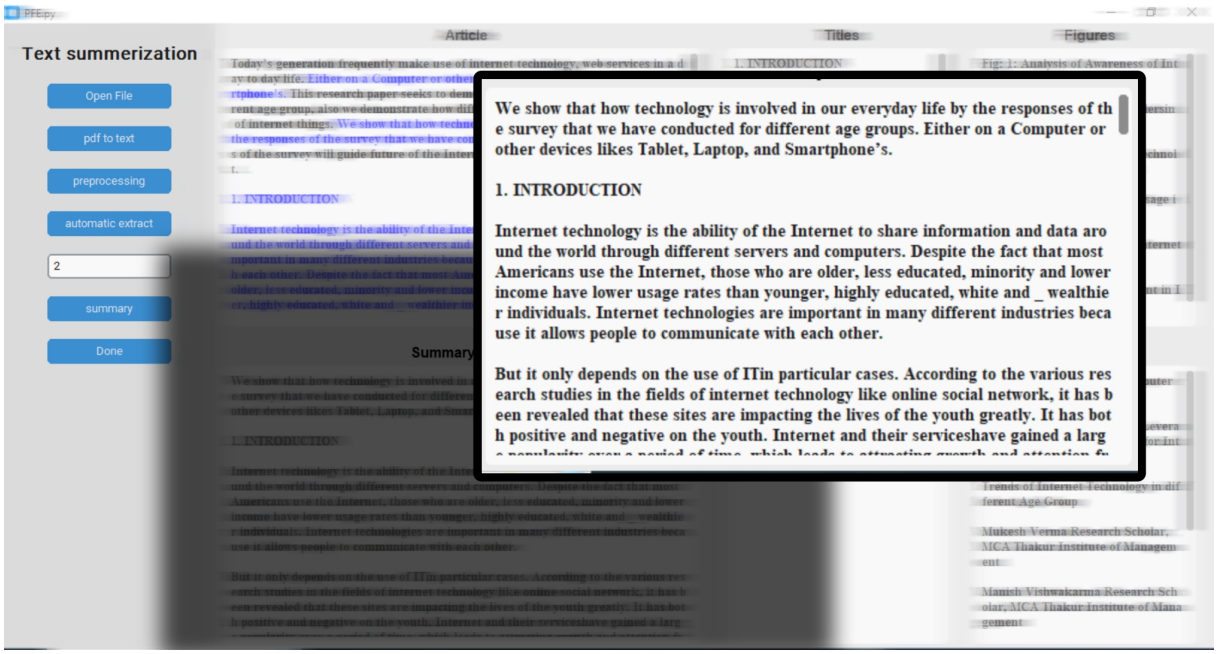

Figure 4.13: Summary generation

8 Results and Discussion

8.1 System evaluation

To assess the performance of our proposed approach, we employed the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [Lin, 2004b] framework. ROUGE includes several measures that automatically evaluate the quality of a summary by comparing it to ideal summaries created by humans.

In our evaluation, we utilized two specific ROUGE measures: rouge-N (rouge-1 and rouge-2) and rouge-L. ROUGE-N focuses on N-gram ($N \geq 1$) recall, which measures the similarity between a system summary and human-generated or reference summaries. It provides an indication of the fluency or coherence of the summaries. The calculation of ROUGE-N is based on Equation 4.2. In our case, we considered N values of 1 and 2. Specifically, ROUGE-1 and ROUGE-2 represent the overlap of 1-grams and bi-grams, respectively, between the system-generated summaries and the sample summaries. he formula for calculating ROUGE-N is given by:

$$\text{ROUGE-N} = \frac{\text{Count of overlapping N-grams}}{\text{Count of total N-grams in the reference summary}} \tag{4.1}$$

In this formula:

- "Count of overlapping N-grams" refers to the number of N-grams (contiguous sequences of N words) that appear in both the system-generated summary and the reference summary.
- "Count of total N-grams in the reference summary" represents the total number of N-grams present in the reference summary.

ROUGE-L is used to identify the longest co-occurring sequence of n-grams between a system-generated summary and a reference summary. It focuses on capturing the common subse-

quences or phrases shared by both summaries. [Mallick *et al.*, 2019] The formula for calculating ROUGE-L is given by:

$$\text{ROUGE-L} = \frac{\text{Length of longest common subsequence of n-grams}}{\text{Length of reference summary}} \tag{4.2}$$

In this formula:

- "Length of the longest common subsequence of n-grams" refers to the length of the longest continuous sequence of n-grams that appear in both the system-generated summary and the reference summary.
- "Length of the reference summary" represents the total length of the reference summary in terms of the number of words or n-grams.

For extracting the F-measure for each ROUGE-1, ROUGE-2, and ROUGE-L, we employed a combination of precision and recall metrics.

$$F = \frac{2 \times (Recall \times Precision)}{Recall + Precision} \tag{4.3}$$

Precision represents the proportion of relevant documents or summaries among the total retrieved or generated ones. It measures the accuracy and correctness of the information presented in the automated summary.

$$Precision = \frac{Correct}{Correct + Wrong} \tag{4.4}$$

On the other hand, recall assesses how much relevant information from the reference summaries is captured in the automated summary, highlighting the system's ability to cover the essential content.

$$Recall = \frac{Correct}{Correct + Missed} \tag{4.5}$$

By employing precision and recall metrics in combination with the F-measure, we ensure a holistic assessment of the summarization system's performance. This allows us to evaluate the quality and relevance of the generated summaries.

8.2 Results analysis

Table 4.2 presents the F-measures achieved by different evaluation measures. The system achieved a total of 72 % for ROUGE-1, 69 % for ROUGE-2, and 67 % for ROUGE-L.

Table 4.2: F score Rouge evaluation results

Article	Rouge-1	Rouge-2	Rouge-L
1	0.7462	0.7319	0.6684
2	0.6302	0.6133	0.5741
3	0.6065	0.5891	0.5780
4	0.6773	0.6616	0.6283
5	0.7658	0.7371	0.7224
6	0.6245	0.6046	0.5800
7	0.8796	0.8696	0.8451
8	0.8522	0.7383	0.8250

By examining the results presented in Table 4.2, we can observe that our approach successfully captures and conveys important information from the article, even when it is embedded within tables or figures. This indicates the robustness and effectiveness of our method in handling diverse data structures and generating coherent and informative summaries.

In addition to the complex data structures like tables and figures, the article we have summarized also contained multiple titles, which posed an additional challenge to our work. The presence of numerous titles within the article increased the complexity of the summarization process.
However, despite this complexity, our approach was successful in summarizing the content of each individual title separately. We were able to extract and condense the relevant information from each title and generate concise summaries that captured the key points and main ideas presented within them.

In addition to the challenges posed by complex data structures and multiple titles, our approach has a specific requirement regarding the structure of the article being summarized. It functions optimally when applied to articles with a two-column or one-column structure.

Another challenge we encountered in our approach is the presence of equations and formulas within the articles we summarized. These mathematical expressions pose a difficulty in the summarization process and can potentially reduce the quality of the generated summaries.

Equations and formulas often contain essential information and contribute significantly to the understanding of the article's content. However, capturing the full meaning and context of these mathematical expressions within a summary can be a complex task.

9 Conclusion

In this chapter, we have explicated the designed system, along with the tools and languages employed in its development. Additionally, we have showcased the diverse interfaces provided by the system

General Conclusion

The increasing volume of textual information generated across diverse fields and domains presents a formidable challenge in effectively managing and extracting valuable insights from this vast data pool. Manual processing and analysis of every document are not only time-consuming but also impractical. Consequently, there is a growing demand for automated systems that can offer concise summaries of text documents, enabling users to swiftly grasp the key points without the need to review the entire content.

To address this challenge, we have implemented a text summarization system that leverages a combination of approaches. One of the key approaches we employ is TF-IDF (Term Frequency-Inverse Document Frequency) combined with cosine similarity. This technique enables us to identify important terms within documents by considering their frequency of occurrence and their relative importance across the entire corpus. By calculating the cosine similarity between document vectors, we can assess the similarity between documents and select representative sentences or passages for summarization.

In our ongoing research, we strive to enhance the capabilities of text summarization by exploring three specific directions:

Multi-document summarization: We aim to extend our system to handle multiple documents, extracting the most relevant and informative content from a collection of texts.

Abstractive summarization: In addition to extractive summarization, we intend to explore abstractive summarization techniques.

Domain-specific summarization: We recognize the significance of domain-specific summarization, where the system tailors the summarization process to specific fields or domains.

Bibliography

[Abdelrahman, 2019] ABDELRAHMAN, M. M. H. M. (2019). *Analyzing robustness of models of chaotic dynamical systems learned from data with Echo state networks*. Thèse de doctorat, Rice University.

[Andhale et Bewoor, 2016] ANDHALE, N. et BEWOOR, L. (2016). An overview of text summarization techniques. pages 1–7.

[Babar et Patil, 2015] BABAR, S. et PATIL, P. D. (2015). Improving performance of text summarization. *Procedia Computer Science*, 46:354–363.

[Baralis *et al.*, 2015] BARALIS, E., CAGLIERO, L., FIORI, A. et GARZA, P. (2015). Mwisum: A multilingual summarizer based on frequent weighted itemsets. *ACM Transactions on Information Systems (TOIS)*, 34(1):1–35.

[Barzilay et McKeown, 2005] BARZILAY, R. et MCKEOWN, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

[BERROUBI *et al.*, 2022] BERROUBI, A., BEN LATRACHE, S. *et al.* (2022). *Vers un Système pour le Résumé Automatique des Textes Arabes*. Thèse de doctorat, UNIVERSITY of M'SILA.

[Bing *et al.*, 2015] BING, L., LI, P., LIAO, Y., LAM, W., GUO, W. et PASSONNEAU, R. J. (2015). Abstractive multi-document summarization via phrase selection and merging. *arXiv preprint arXiv:1506.01597*.

[Boughaba *et al.*, 2017] BOUGHABA, M., BOUKHRIS, B. et MEFLAH, M. (2017). Lapprentissage profond (deep learning) pour la classification et la recherche d'images par le contenu.

[Chapman et Stolee, 2016] CHAPMAN, C. et STOLEE, K. T. (2016). Exploring regular expression usage and context in python. *In Proceedings of the 25th International Symposium on Software Testing and Analysis*, pages 282–293.

[Cheng *et al.*, 2013] CHENG, K., LI, Y. et WANG, X. (2013). Single document summarization based on triangle analysis of dependency graphs. *In 2013 16th International Conference on Network-Based Information Systems*, pages 38–43. IEEE.

[Divya et Aiswarya, 2021] DIVYA, P. et AISWARYA, V. (2021). Deep learning: Techniques and applications. *Journal of Emerging Technologies and Innovative Research (JETIR)*, 8(7): a125–a129.

[Embar *et al.*, 2013] EMBAR, V. R., DESHPANDE, S. R., VAISHNAVI, A., JAIN, V. et KALLIMANI, J. S. (2013). saramsha-a kannada abstractive summarizer. *In 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 540–544. IEEE.

[Ferreira *et al.*, 2014] FERREIRA, R., de SOUZA CABRAL, L., FREITAS, F., LINS, R. D., de FRANÇA SILVA, G., SIMSKE, S. J. et FAVARO, L. (2014). A multi-document summarization system based on statistics and linguistic treatment. *Expert Systems with Applications*, 41(13):5780–5787.

[García-Hernández et Ledeneva, 2009] GARCÍA-HERNÁNDEZ, R. A. et LEDENEVA, Y. (2009). Word sequence models for single text summarization. *In 2009 Second International Conferences on Advances in Computer-Human Interactions*, pages 44–48. IEEE.

[Geetha et Deepamala, 2015] GEETHA, J. K. et DEEPAMALA, N. (2015). Kannada text summarization using latent semantic analysis. *In 2015 International conference on advances in computing, communications and informatics (ICACCI)*, pages 1508–1512. IEEE.

[Genest et Lapalme, 2011] GENEST, P.-E. et LAPALME, G. (2011). Framework for abstractive summarization using text-to-text generation. *In Proceedings of the workshop on monolingual text-to-text generation*, pages 64–73.

[Genest et Lapalme, 2012] GENEST, P.-E. et LAPALME, G. (2012). Fully abstractive approach to guided summarization. *In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 354–358.

[Greenbacker, 2011] GREENBACKER, C. (2011). Towards a framework for abstractive summarization of multimodal documents. *In Proceedings of the ACL 2011 Student Session*, pages 75–80.

[Hannah *et al.*, 2011] HANNAH, M. E., GEETHA, T. et MUKHERJEE, S. (2011). Automatic extractive text summarization based on fuzzy logic: a sentence oriented approach. *In Swarm, Evolutionary, and Memetic Computing: Second International Conference, SEMCCO 2011, Visakhapatnam, Andhra Pradesh, India, December 19-21, 2011, Proceedings, Part I 2*, pages 530–538. Springer.

[Harabagiu et Lacatusu, 2002] HARABAGIU, S. M. et LACATUSU, F. (2002). Generating single and multi-document summaries with gistexter. *In Document Understanding Conferences*, pages 11–12.

[Hardeniya *et al.*, 2016] HARDENIYA, N., PERKINS, J., CHOPRA, D., JOSHI, N. et MATHUR, I. (2016). *Natural language processing: python and NLTK*. Packt Publishing Ltd.

[Harmaakivi, 2022] HARMAAKIVI, R. (2022). Streamlining the pdf translation process with a python application.

[He *et al.*, 2008] HE, T., LI, F., SHAO, W., CHEN, J. et MA, L. (2008). A new feature-fusion sentence selecting strategy for query-focused multi-document summarization. *In 2008 International Conference on Advanced Language Processing and Web Information Technology*, pages 81–86. IEEE.

[Hingu *et al.*, 2015] HINGU, D., SHAH, D. et UDMALE, S. S. (2015). Automatic text summarization of wikipedia articles. *In 2015 international conference on communication, information & computing technology (ICCICT)*, pages 1–4. IEEE.

[Hinton, 2007] HINTON, G. E. (2007). Boltzmann machine. *Scholarpedia*, 2(5):1668.

[Huang *et al.*, 2019] HUANG, X., HONG, S. H., YU, M., DING, Y. et JIANG, J. (2019). Demand response management for industrial facilities: A deep reinforcement learning approach. *IEEE Access*, 7:82194–82205.

[Islam, 2015] ISLAM, Q. N. (2015). *Mastering PyCharm*. Packt Publishing Ltd.

[Jayashree *et al.*, 2012] JAYASHREE, R., SRIKANTA, M. K. et ANAMI, B. S. (2012). Categorized text document summarization in the kannada language by sentence ranking. *In 2012 12th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 776–781. IEEE.

[Kågebäck *et al.*, 2014] KÅGEBÄCK, M., MOGREN, O., TAHMASEBI, N. et DUBHASHI, D. (2014). Extractive summarization using continuous vector space models. *In Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39.

[Kaikhah, 2004] KAIKHAH, K. (2004). Automatic text summarization with neural networks. *In 2004 2nd International IEEE Conference on'Intelligent Systems'. Proceedings (IEEE Cat. No. 04EX791)*, volume 1, pages 40–44. IEEE.

[Kantzola, 2020] KANTZOLA, E. (2020). Extractive text summarization of greek news articles based on sentence-clusters.

[Khurana *et al.*, 2023] KHURANA, D., KOLI, A., KHATTER, K. et SINGH, S. (2023). Natural language processing: State of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744.

[Kuppan et Sobha, 2009] KUPPAN, S. et SOBHA, L. (2009). An approach to text summarization. *In Proceedings of the Third International Workshop on Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies (CLIAWS3)*, pages 53–60.

[Lee *et al.*, 2005] LEE, C.-S., JIAN, Z.-W. et HUANG, L.-K. (2005). A fuzzy ontology and its application to news summarization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(5):859–880.

[Li *et al.*, 2018] LI, J., XI, B., LI, Y., DU, Q. et WANG, K. (2018). Hyperspectral classification based on texture feature enhancement and deep belief networks. *Remote Sensing*, 10(3):396.

[Liddy, 2001] LIDDY, E. D. (2001). Natural language processing.

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

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

[Lin et Och, 2004] LIN, C.-Y. et OCH, F. (2004). Looking for a few good metrics: Rouge and its evaluation. *In Ntcir workshop*.

[Luhn, 1957] LUHN, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317.

[Malliaros et Skianis, 2015] MALLIAROS, F. D. et SKIANIS, K. (2015). Graph-based term weighting for text categorization. *In Proceedings of the 2015 IEEE/ACM international conference on advances in social networks analysis and mining 2015*, pages 1473–1479.

[Mallick *et al.*, 2019] MALLICK, C., DAS, A. K., DUTTA, M., DAS, A. K. et SARKAR, A. (2019). Graph-based text summarization using modified textrank. *In Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018*, pages 137–146. Springer.

[Mihalcea, 2004] MIHALCEA, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. *In Proceedings of the ACL interactive poster and demonstration sessions*, pages 170–173.

[Moawad et Aref, 2012] MOAWAD, I. F. et AREF, M. (2012). Semantic graph reduction approach for abstractive text summarization. *In 2012 Seventh International Conference on Computer Engineering & Systems (ICCES)*, pages 132–138. IEEE.

[Modaresi et Conrad, 2014] MODARESI, P. et CONRAD, S. (2014). From phrases to keyphrases: An unsupervised fuzzy set approach to summarize news articles. *In Proceedings of the 12th International Conference on Advances in Mobile Computing and Multimedia*, pages 336–341.

[MOHAMMED CHIKOUCHE, 2017] MOHAMMED CHIKOUCHE, S. (2017). *Résumé automatique des textes*. Thèse de doctorat, Faculté des Mathématiques et de l'Informatique-Université Mohamed BOUDIAF-M'sila.

[Mridha *et al.*, 2021] MRIDHA, M. F., LIMA, A. A., NUR, K., DAS, S. C., HASAN, M. et KABIR, M. M. (2021). A survey of automatic text summarization: Progress, process and challenges. *IEEE Access*, 9:156043–156070.

[Nikolov, 2020] NIKOLOV, N. I. (2020). *Abstractive Document Summarization in High and Low Resource Settings*. Thèse de doctorat, ETH Zurich.

[Oliphant *et al.*, 2006] OLIPHANT, T. E. *et al.* (2006). *A guide to NumPy*, volume 1. Trelgol Publishing USA.

[Ozsoy *et al.*, 2011] OZSOY, M. G., ALPASLAN, F. N. et CICEKLI, I. (2011). Text summarization using latent semantic analysis. *Journal of Information Science*, 37(4):405–417.

[Papineni *et al.*, 2002] PAPINENI, K., ROUKOS, S., WARD, T. et ZHU, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. *In Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

[Prakash et Shukla, 2014] PRAKASH, C. et SHUKLA, A. (2014). Human aided text summarizer" saar" using reinforcement learning. *In 2014 International Conference on Soft Computing and Machine Intelligence*, pages 83–87. IEEE.

[Ragunath et Sivaranjani, 2015] RAGUNATH, R. et SIVARANJANI, N. (2015). Ontology based text document summarization system using concept terms. *ARPN J Eng Appl Sci*, 10(6): 1819–660.

[Ray *et al.*, 2020] RAY, S., ALSHOUILIY, K. et AGRAWAL, D. (2020). Dimensionality reduction for human activity recognition using google colab. *Information*, 12:6.

[Romero, 2017] ROMERO, J. (2017). Abstractive text summarisation with neural networks. *no. January*, 2017.

[Saggion *et al.*, 2002] SAGGION, H., RADEV, D., TEUFEL, S. et LAM, W. (2002). Meta-evaluation of summaries in a cross-lingual environment using content-based metrics. *In COLING 2002: The 19th International Conference on Computational Linguistics*.

[Sarkar, 2012] SARKAR, K. (2012). An approach to summarizing bengali news documents. *In proceedings of the International Conference on Advances in Computing, Communications and Informatics*, pages 857–862.

[Sarkar *et al.*, 2011] SARKAR, K., NASIPURI, M. et GHOSE, S. (2011). Using machine learning for medical document summarization. *International Journal of Database Theory and Application*, 4(1):31–48.

[Seetha *et al.*, 2023] SEETHA, H., TIWARI, V., ANUGU, K. R., MAKKA, D. et KARNATI, D. (2023). A gui based application for pdf processing tools using python & customtkinter. *Int. J. Res. Appl. Sci. Eng. Technol.*

[Sharma *et al.*, 2022] SHARMA, V., JAIN, M., JAIN, T. et MISHRA, R. (2022). License plate detection and recognition using opencv–python. *In Recent Innovations in Computing: Proceedings of ICRIC 2021, Volume 1*, pages 251–261. Springer.

[Smith, 2007] SMITH, R. (2007). An overview of the tesseract ocr engine. *In Ninth international conference on document analysis and recognition (ICDAR 2007)*, volume 2, pages 629–633. IEEE.

[Steinberger *et al.*, 2004] STEINBERGER, J., JEZEK, K. *et al.* (2004). Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM*, 4(93-100):8.

[Tanaka *et al.*, 2009] TANAKA, H., KINOSHITA, A., KOBAYAKAWA, T., KUMANO, T. et KATO, N. (2009). Syntax-driven sentence revision for broadcast news summarization. *In Proceedings of the 2009 Workshop on Language Generation and Summarisation (UCNLG+ Sum 2009)*, pages 39–47.

[Thu *et al.*, 2013] THU, H. N. T., HUU, Q. N. et NGOC, T. N. T. (2013). A supervised learning method combine with dimensionality reduction in vietnamese text summarization. *In 2013 Computing, Communications and IT Applications Conference (ComComAp)*, pages 69–73. IEEE.

[Thu et Ngoc, 2014] THU, H. N. T. et NGOC, D. V. T. (2014). Improve bayesian network to generating vietnamese sentence reduction. *IERI Procedia*, 10:190–195.

[Umesh, 2012] UMESH, P. (2012). Image processing in python. *CSI Communications*, 23(2).

[Van Rossum et Drake Jr, 1995] VAN ROSSUM, G. et DRAKE JR, F. L. (1995). *Python tutorial*, volume 620. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands.

[Varadarajan et Hristidis, 2006] VARADARAJAN, R. et HRISTIDIS, V. (2006). A system for query-specific document summarization. *In Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 622–631.

[Verspoor et Cohen, 2013] VERSPOOR, K. et COHEN, K. B. (2013). Natural language processing. *Encyclopedia of Systems Biology*, pages 1495–1498.

[Wasson, 1998] WASSON, M. (1998). Using leading text for news summaries: Evaluation results and implications for commercial summarization applications. *In COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*.

[Wu *et al.*, 2015] WU, K., SHI, P. et PAN, D. (2015). An approach to automatic summarization for chinese text based on the combination of spectral clustering and lexrank. *In 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 1350–1354. IEEE.

[Xu *et al.*, 2016] XU, W., NAPOLES, C., PAVLICK, E., CHEN, Q. et CALLISON-BURCH, C. (2016). Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

[Yadav *et al.*, 2022] YADAV, D., DESAI, J. et YADAV, A. K. (2022). Automatic text summarization methods: A comprehensive review. *arXiv preprint arXiv:2204.01849*.

[Zhang et Li, 2009] ZHANG, P.-y. et LI, C.-h. (2009). Automatic text summarization based on sentences clustering and extraction. *In 2009 2nd IEEE international conference on computer science and information technology*, pages 167–170. IEEE.