Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (34^{th}cycle)

Deep Learning for Natural Language Understanding and Summarization

Moreno La Quatra

******

Supervisor(s):
Prof. Luca Cagliero, Supervisor

Doctoral Examination Committee:
Prof. Di Caro Luigi , Referee, Università degli studi di Torino
Prof. Quintarelli Elisa, Referee, Università degli studi di Verona
Prof. Campos Ricardo, Instituto Politécnico De Tomar
Prof. Mellia Marco, Politecnico di Torino
Prof. Papotti Paolo, École Nationale Supérieure des Télécommunications

Politecnico di Torino
2022

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Moreno La Quatra
2022

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

To Viviana, who was there every step of the way, and to my family, who never stopped believing in me.

Acknowledgements

I want to take this opportunity to thank everyone who has contributed to the success of this research, be they through their efforts, advice, help, support, or friendship.

First, I would like to express my gratitude to my supervisor Prof. Luca Cagliero for the innumerable hours of stimulating discussion, advice, and encouragement which have made this research possible. I would also like to thank the reviewers of this thesis, Prof. Luigi Di Caro and Prof. Elisa Quintarelli, for their insightful and constructive comments that helped to further improve the quality of this work.

Additionally, I want to thank all the researchers of the DataBase and Data Mining Group at Politecnico di Torino with whom I had the pleasure of collaborating on many projects and publications. In particular, I want to thank Prof. Paolo Garza, Prof. Elena Baralis, and Prof. Daniele Apiletti for their advice and support during my PhD.

I'd also like to thank all my colleagues at Lab 5. Each one of them has contributed in his or her own way to the creation of a stimulating working environment in which I have always enjoyed working. The PhD journey has been long and challenging, but the coffee breaks, lunches, and office discussions have made it an unforgettable experience.

My family has always been very supportive and I am grateful to my parents Paola and Giuseppe, my sisters Giulia and Greta and my aunt Benedetta for always believing in me and for their constant encouragement, even when I was far away.

Last but not least, I want to thank the most important person in my life, my partner Viviana. Her unconditional love, support, and patience in the good and the bad moments, have been and continue to be the most precious gift I have ever received.

Abstract

Information overload is a major problem affecting today's society. The continuous stream of information makes it impossible for individuals to process and understand the vast amount of information available to them. Automatic content understanding is a research field that aims to alleviate this problem by efficiently extracting relevant information from unstructured sources (e.g., text documents). Natural Language Processing (NLP) is a branch of artificial intelligence involving automatic interpretation and manipulation of human language. The use of NLP techniques is key to many automatic content understanding tasks, such as information extraction and text summarization.

Our research aims at investigating and developing methods for automatic content understanding to alleviate the effects of information overload. We focus on three specific data types: academic articles, news stories, and podcasts. In order to automate the extraction of relevant information from unstructured sources, we explore and present several methodologies tailored to each domain.

Our contributions in the field of scientific literature analysis aim at providing researchers with resources to navigate the ever-growing amount of scientific papers. First, we analyze citation context to assess whether reading the full text of a publication is likely to be useful for understanding the referencing paper, thereby reducing the amount of time spent on literature search. Second, we devise two different models that process scientific papers to extract a results-oriented overview of its main contributions (namely, the highlights) and facet-specific summaries tailored to the needs of different types of readers. Finally, we explore the potential of unsupervised summarization models to generate slides for scientific talks, thus providing authors with an initial draft of their presentation.

As part of our work in the news domain, we concentrate on the task of news timeline generation, a process for automatically creating a timeline of specific

events that summarizes the most important developments that happened during event unfolding. We present a pipeline that summarizes what happened at each date and then selects salient dates according to graph-based centrality measures. Since the methodology is specifically tailored to the English language, we separately investigate the same problem in multilingual settings and propose a solution that, given a set of documents in multiple languages, incorporates cross-lingual alignment and automatic machine translation to generate timelines in a target language.

Finally, we investigate the challenge of podcast summarization and propose an extractive summarization approach that leverages multimodal information (i.e., audio and text) to identify the most important segments of an episode. Our system relies on an end-to-end deep learning architecture that jointly learns feature representations from both modalities. It combines text and audio features using a late-fusion strategy and is trained to score and select relevant segments from the spoken content. By automatically selecting and concatenating the most important portions, this methodology can provide both a textual and an audio summary of a podcast episode. Our system is further enhanced by proposing a select-and-rewrite approach that can generate a more fluent summary by rearranging and rewriting selected segments using a state-of-the-art sequence-to-sequence model.

The research domains investigated in this thesis share the need to manipulate linguistic information and, more specifically, human language. Therefore, we extensively use NLP techniques to understand the semantics of language and automatically process unstructured content. Considering a broad range of scenarios, this thesis examines how NLP can enhance the understanding of human language and facilitate the processing of large volumes of data.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Natural Language Processing (NLP) is the research field at the intersection of linguistics, Artificial Intelligence, and computer science. Its goal is to automate human language comprehension, understanding, and generation. People use natural language to communicate, which consists of a finite set of symbols (letters, words, punctuation marks, etc.) that are arranged according to grammar rules. Essentially, NLP is the development of computational models that can automatically understand natural language text, i.e., to extract meaning from text.

The volume of written content on the World Wide Web has grown dramatically over the past few years, making NLP an increasingly important field. A vast amount of text is generated and consumed by personal computers, smartphones, and smart devices, making it difficult for humans to read and process all of this information. The Internet has fundamentally changed the way we communicate, consume information, and work. It has established the transition into the information age [20], where we are overwhelmed with textual data that requires processing and analysis. Companies' and economies' competitiveness is strongly dependent on access to information and, more importantly, on the knowledge that can be extracted from such data. NLP is the key to unlocking the value in this data by providing the ability to automatically extract meaning and understanding from text.

Access to information used to be time-consuming and difficult, but with the advent of the Internet, we are now able to access to a wide range of information in seconds. Despite this is an undeniable advancement for individuals, businesses, and societies, it has also brought a new challenge: information overload.

Information overload is the condition of having too much information to effectively process it. It refers to the challenges that arise when a large amount of information is available, and it is difficult to filter the relevant information. Alvin Toffler was the first to propose this theory [179] and the importance of this phenomenon has grown in importance in recent years due to the exponential growth of the Internet and the amount of data available online.

Information overload may have negative effects on productivity, decision making, and overall well-being [60]. It can also lead to suboptimal decisions, as people are unable to process all the available information and identify the most relevant one. Information retrieval and NLP can help to mitigate this problem by providing methods to effectively filter and process information. Specifically, the recent advances in deep learning have enabled the development of models that are able to automatically understand and process textual data, and thus, can be used to address this problem.

Deep learning, however, is not limited to NLP applications and has been successfully applied to a variety of other domains and data modalities. In the field of computer vision, for example, it has led to the development of applications such as facial recognition and object classification by automatically learning to extract relevant features from images. Similarly, in the field of NLP, it has led to applications covering text classification or summarization which rely on the understanding of language to automatically process text data.

Deep learning methodologies make it possible to develop models that automatically extract relevant features from data. These models can be trained on large databases without having to rely on human supervision. This allows for the development of models that are able to generalize to new data. In NLP, the use of self-supervised techniques (e.g., training objectives that do not require human-annotated data) has led to the development of pre-trained language models that are able to provide reliable starting point for many natural language understanding tasks. By fine-tuning the model on a small dataset, transfer learning allows the knowledge learned from a pre-trained model to be applied to another task. As a result, these pre-trained models can be used to build a wide variety of applications with a limited amount of data.

The joint use of self-supervision and transfer learning makes it possible to automatically understand written content and alleviate the problem of information overload by efficiently extracting relevant information from large volumes of data.

News articles, for example, often contain a large amount of information that may be irrelevant to a particular reader. A system capable of understanding content can automatically filter news articles to only present the most relevant ones to the reader, which allows for a better focus on the most relevant information. Similarly, long documents such as reports or research papers can be summarized to reduce the amount of information that needs to be processed by the reader. Those approaches require both the ability to read and understand the text, as well as the ability to identify the most relevant information.

Most of the contributions discussed in this thesis focus on the task of content summarization, which entails the generation of a summary that contains the most important information from a given text. Automatic summarization is essential for mitigating information overload, since it can enable users to quickly read short summaries of long documents. The dissertation discusses how deep learning can address the challenges that may arise when applying automatic summarization to different domains and provides a detailed evaluation of the proposed solutions. It presents a thorough exploration of several application domains, including scientific document analysis, temporal event summarization, and spoken language understanding.

This dissertation focuses on the following research objectives:

- Examine how traditional machine learning techniques and modern deep learning approaches can be applied to extract and summarize textual and multimodal content.
- Analyze the benefits of leveraging the semantic representations learned by deep learning models for content summarization.
- Develop models that can effectively utilize temporal and multilingual information for content summarization.
- Explore different application domains and conduct a comprehensive evaluation of the proposed methods.

Through these objectives, we propose a set of models that can effectively summarize textual and multimodal content, and demonstrate their effectiveness across multiple fields. The application domains considered in the present dissertation encompass:

- *Scientific publications*: summarizing a scientific paper requires an in-depth analysis of the document structure and citation-level exploration of its content.
- *News summarization*: summarizing news articles poses a number of challenges due to (i) the rapid development of newsworthy events, which calls for monitoring the news timeline and related aspects; (ii) the presence of global data sources, which require effective cross-lingual information retrieval and analysis.
- *Spoken-content summarization*: the summarization of speech content (e.g., podcasts) calls for effective multimodal modeling, as it requires the joint analysis of information from acoustic and textual sources.

1.1 Dissertation outline

This thesis is organized as follows. Chapter 2 introduces the main contributions of the thesis, analyzing the main challenges and providing an overview of the different application domains where machine learning and deep learning can be used for content understanding and summarization.

Detailed information about the methodologies used in this dissertation is presented in Chapter 3, which covers the background of deep learning and NLP. These concepts include word embeddings, neural architectures, and sequence-to-sequence models, which form the foundation for deep learning in this field.

The task of scientific literature analysis is the focus of Chapter 4. It introduces the domain of scientometrics and explains the different types of data that can be used to analyze scientific literature. It then describes the contributions of this thesis in this field and shows how deep learning models can be used to analyze the citation context and extract summaries from research papers.

Chapter 5 discusses the importance of time in text analysis and how to model temporal information that is present in text data. It introduces the task of news timeline generation along with the solutions proposed in both single-language and multilingual settings.

Chapter 6 focuses on the task of comprehending spoken content from a multimodal perspective. It discusses how text and audio modalities can be combined together to understand spoken content. The chapter also outlines the task of spoken content summarization and describes how deep learning models can be used to leverage both textual and acoustic features to summarize spoken content.

Each chapter also discusses the future directions in its respective field, highlighting both the challenges and the opportunities that lie ahead. Finally, Chapter 7 concludes the dissertation by summarizing the main findings and discussing general recommendations for future research.

Chapter 2

Research contributions

This chapter introduces the application domains and the research contributions of this thesis. The ubiquitous nature of text data has led to the analysis of natural language in a variety of contexts, such as scientific documents, news articles, and spoken content. While each of these domains presents unique challenges, they all share the common goal of extracting and summarizing the most important information. Indeed, extracting or generating summaries can be beneficial across all of these areas, as it can enable users to quickly obtain an overview of the most valuable information.

2.1 Scientific literature analysis

Over the past few years, the amount of scientific literature has increased dramatically. This makes it challenging for researchers to keep up to date with the latest advances in their field. The introduction of digital libraries and online journals has simplified the access to scientific literature, but it has also resulted in more information that researchers need to process. In parallel, open-access journals and pre-print servers (e.g., ArXiv[1] or PubMed [2]) have greatly sped up the process of paper submission and publication. Consequently, there is a constant stream of new papers, which makes keeping up with the latest advances even more challenging for researchers.

Similar to other fields, the increasing amount of scientific literature has contributed to information overload. Hence, text mining algorithms have been designed

[1] www.arxiv.org
[2] https://www.ncbi.nlm.nih.gov/pmc/

to process large collections of scientific papers to detect relevant patterns and trends. These methods can be used to identify new research directions, find hidden patterns, or identify potential collaborators.

The task of automatic text summarization is a useful tool for researchers who would like to stay up-to-date with their field's latest developments. Particularly, the task of summarizing scientific papers aims at creating a short summary that contains the most significant information from a paper. Researchers can use this summary to quickly ascertain whether a paper is relevant to their interests, as well as provide a starting point for further reading. Automatic content understanding is a key tool for helping researchers to focus on the most relevant papers and for identifying emerging topics. Text summarization can be tailored to the needs of researchers, and can be used to generate summaries that highlight the most important findings of a scientific publication.

It is essential to recognize that scientific papers have highly technical language and formal structures, thus affecting the challenges automatic text summarization systems must overcome. In contrast to general-purpose texts, scientific papers are usually long, contain specialized terminology, and have a strict structure that includes elements such as the abstract, introduction, methods, results, and discussion. Various readers may be interested in different sections as each of them may present different information. The abstract, for example, may not be enough to provide a full understanding of the paper, whereas the discussion may provide information that cannot be found in the abstract. A reader that is interested in exploring the main findings of a paper may be interested in having a quick overview of the results, whereas a reader that wants to reproduce the outcomes of the work may prefer to have a summary that includes the details of the methodology.

The variability of information found within a scientific paper, combined with the different needs of the reader, make it challenging to summarize scientific papers, which requires an understanding of the underlying semantics. By leveraging deep learning techniques we investigate how content understanding methodologies can be used for scientific literature analysis.

- Using a semantically-grounded approach we analyze and classify the citation context of scientific papers to discover whether the full text of the referenced paper is worth reading, based on the expectation that the full text contains additional information related to the citation context.

- We present a methodology that is specifically tailored to extract sentences that describe the main results and/or contributions of a scientific paper.
- We build a model that generates different summaries covering specific aspects of scientific papers. The methodology allows for a fine-grained analysis of the publications by analyzing the discourse facets and exploiting their underlying structure.

The in-depth analysis of the content and the format of scientific papers can help us better understand the research itself. Through this understanding, we can increase the efficiency of literature searches and create summaries that will be beneficial to the scientific community, providing them with better knowledge discovery tools.

2.2 The importance of time in text analysis

In the current landscape of text analysis, the study of temporal aspects of texts is of utmost importance. As online news sites are constantly publishing a stream of articles, issues raised by information overload and the proliferation of online content are increasingly relevant to online journalism. Understanding the evolution of content over time is critical for establishing the context of events, and news timelines can provide valuable insight into how events are evolving over time.

Similarly to academic articles, the volume of online news makes it difficult for readers to identify the most important and interesting stories. The average person is not able to read all the news pertaining to a particular topic and must prioritize which articles to read. Automatic content understanding allows for identification of the most important news stories, as well as modeling the evolving context of events.

Long-term events, such as the evolution of a disease outbreak or the progress of a political campaign, require the ability to track the evolution of the event over time. Early in the unfolding of events, we are often able to gather the most pertinent information, so it is critical to identify and track these developments at this stage. An overall timeline can provide a valuable overview of the event's development, and can be used to identify key turning points or major milestones.

Considering time-related information is therefore critical for understanding and analyzing the evolution of events reported in online news articles. We propose a new technique for detecting salient dates associated with a specific event, and then using these dates to summarize how the event evolved over time. We also explored the use of multilingual resources for this task and investigated the potential of cross-lingual summarization for events reported in multiple languages.

2.3 Multimodal content understanding

Although text still is the most common form of online communication, spoken content (such as podcasts and videos) is also gaining in popularity. Users can be interested in spoken content for a variety of reasons, including to obtain information about a particular topic, to entertain themselves, and to learn something of interest. Many platforms host large amounts of spoken content. Following the same pattern as with text, it is becoming more and more difficult for users to identify the most worthwhile information. It can be helpful to quickly identify the most relevant content in order to mitigate the adverse effects of information overload. Hence, users need a way of identifying the most relevant content without having to listen to the entire podcast or watch the entire video.

As spoken content is typically less structured than text, its automatic analysis presents several challenges. The audio material is diverse as it includes news programs, interviews, speeches, and lectures, as well as multimodal (e.g., mainly text and audio). The conventional approaches to analyzing speech are based on a two-step process. In the first step, the audio is transcribed into text and in the second step it is analyzed using NLP techniques. The automatic transcription of audio content, however, is a challenge in itself, as transcription quality is rarely completely accurate. The resulting errors can introduce noise into the automatic analysis of spoken content. For instance, automated transcription may not identify key words in specific domains and may introduce grammatical errors, thus confusing the interpretation of the text.

Analyzing only the transcriptions of audio content is also problematic because it reduces the analysis to one modality, whereas spoken content is often multimodal. Speech contains significant information not only through the words that are spoken, but also through other modalities, such as intonation and pauses. The emotions conveyed by speakers through their tone and nonverbal communication can provide helpful contextual information. Whenever the analysis is limited to the words spoken in the audio, this information is lost. As part of our contribution in this field, we combine information from transcriptions and audio signals to summarize speech recordings. We aim to enhance the automatic analysis of spoken content by leveraging the multimodal information in the audio signal. The analysis focuses on podcast episodes, a form of spoken content that is gaining popularity over the last few years.

Podcasts are audio files that can be downloaded from the Internet and played on smartphones or computers. They are typically episodic, with each episode covering a specific topic. Podcast episodes are typically less structured than written documents, and their length can vary (e.g., from minutes to several hours). Adding an automatic summary to each podcast episode makes it easier for users to quickly identify key points and decide whether they want to listen to the entire episode.

Chapter 3

Background

Natural language is the primary mean we use to communicate and express our thoughts. The purpose of this chapter is to give a brief overview of the field of NLP and, more broadly semantic content understanding, from the early days to the current state of the art, with a focus on the deep learning-based approaches. Section 3.1 reviews the history of NLP and discusses the fundamentals of statistical and neural-based approaches. The most important findings for the design of neural language models, as well as their implication for the research topics discussed in this thesis are summarized in Section 3.2 and a specific focus on language understanding and generation is provided in Section 3.2.4 and Section 3.2.5, respectively. Finally, an in-depth analysis of the text summarization task is provided in Section 3.3.

3.1 Natural Language Processing: A brief history

The end goal of NLP is to let machines be able to understand text written by humans in natural language. The academic interest in NLP can be traced back to the 1950s when the mathematician Alan Turing proposes the *imitation game*, a test to evaluate the ability of a given machine to *exhibit intelligent behavior* [183]. The test was structured as a conversation, with questions formulated by humans in natural language and responses generated by an automated system that should be able to (i) understand the semantics of the conversation and (ii) interact in natural language (i.e., English). Since then, the research community has investigated different approaches to let a machine *understand* natural language and *generate* text similar to human beings.

During the 1960s, the research community focused on generating handwritten rules for modeling human language (primarily English). At that time, data-driven approaches were not considered practical due to the high overhead of processing, the large amount of data needed, and the lack of effective learning algorithms. Manually-defined rules were the only viable option for modeling language. They were formulated to take into account local linguistic dependencies in specific NLP tasks, e.g., Machine Translation [192]. However, those approaches had some major drawbacks. They heavily relied on the expertise of the rule designers and were often difficult to maintain and extend to new languages. Moreover, the rules failed to capture the global statistical patterns in language, crucial for many NLP tasks.

It was only in the late 1980s that statistical methods pushed the field forward, thanks to the increasing computational power of computers and the availability of large text corpora. They overcome the limitations of previous approaches and enable the modeling of long dependencies in language while automating the learning process and minimizing the need for manual effort [108]. This fundamental change from traditional to statistical methods laid the foundations for the design of new machine learning algorithms based on Decision Trees [146] or Hidden Markov Models [7] that have been successfully applied in several NLP tasks such as part-of-speech tagging [35] and coreference resolution [116].

While applying statistical methods, language processing and generation tasks were based on the concept of *n-grams*. An n-gram is defined as a contiguous sequence of *n* items from a given text snippet. They can be defined in terms of words or characters. For example, a word-based bi-gram (where *n* is set to 2) for the sentence *"natural language processing was born in the 1950s"* will include *"natural language"*, *"language processing"*, *"processing was"* and so on. Similarly, a char-based bi-gram will be *"na"*, *"at"*, *"tu"*, *"ur"*, *"ra"*, *"al"* and so on. In both cases, the idea is to take a sequence of items and apply statistical methods to find out which of them are likely to occur together. This simple yet effective definition allowed the design of new innovative systems in the field of text mining [174], information retrieval [125] and, text generation [113] to name a few. Leveraging the n-gram definitions it was possible to define new way for representing documents according to their constituents' words.

Bag-of-words text vectorization The process of mapping text into vectors is known as text vectorization. Early approaches in this domain leverage the co-occurrence of words inside the text to generate a vector representation. This method is known as *bag-of-words*. Given a document D containing a given sequence of words $(w_1, w_2, ...w_n) \in D$, the bag-of-words representation of D corresponds to a vector where each dimension refers to a specific word w_i. Each vector cell is set to 1 if the corresponding word appears in the document while is set to 0 otherwise. Although those approaches are used in some of the more recent commercial systems, they still have major disadvantages. First, it is challenging to manipulate vectors because of their high dimensionality. Second, the vectors are sparse, meaning many cells are set to 0, which makes computation difficult and memory-intensive. Lastly, the vectors can not encode contextual information; hence, words that are related to each other, within the same context, are not represented by similar vectors.

Vector Space Model (VSM) A fundamental problem of the bag-of-words approach is the binary encoding of the presence of a word in a document. While this vectorization technique is straightforward to use, it does not encode the frequency of a word in a document. This issue motivated the development of the *Vector Space Model* (VSM) [162]. It was originally proposed in the context of information retrieval and has been extensively used in several NLP applications. With the VSM approach, each word in a document is encoded according to its frequency, rather than its presence. Similarly to the previous case, given a document D containing a given sequence of words $(w_1, w_2, ...w_n) \in D$, the representation of D corresponds to a vector where each dimension refers to a specific word w_i. In its original formulation, each vector cell is set to the frequency of the corresponding word inside the document. This allows for a more accurate encoding of the documents and can be seen as a generalization of the bag-of-words approach. This methodology has also been extended by replacing the word frequency with the *term frequency-inverse document frequency* (TF-IDF) [161] that adds another term to the word frequency, the inverse document frequency (IDF), in order to down-weight words that appear in many documents (e.g., words that are very common in the collection, thus, less discriminative). Even though this approach is more accurate than bag-of-words, it still fails to address other issues related to sparse vectors, high dimensionality, and the inability to encode context information.

Latent Semantic Analysis (LSA) *Latent Semantic Analysis* (LSA) [37] is a statistical technique that has found a variety of applications in NLP, including text clustering and classification, query expansion, and information retrieval. The LSA approach is based on the Singular Value Decomposition (SVD) [53], a linear algebra technique that is used to compress a matrix and to provide the latent semantic relationships of the data. In this context, the matrix is the so-called *term-document matrix*, where each row corresponds to a term in the vocabulary and each column corresponds to a document in the collection. LSA serves as a text vectorization technique for representing documents in a lower-dimensional space. The new representation is based on the SVD of the term-document matrix, which is used to find the latent relationships among the terms and the documents. In this way, the information that is contained in the high-dimensional matrix is condensed into a lower-dimensional space, and the documents are represented by linear combinations of the latent *concepts* that have been extracted from the original matrix. In contrast to bag-of-words and VSM, LSA does not consider words in isolation, but instead, considers the relationships among them. It overcomes the issues of data sparsity and high dimensions by creating a new, lower-dimensional space that contains the latent concepts of the original sparse matrix. While LSA overcomes the sparsity issue it still fails to address some of the problems related to the previous approaches. It does not consider the context and the word order, thus, while it can leverage the relationships among words, it cannot identify their semantic meaning and does not consider their sequential order.

With the introduction of deep learning in the early 2000s, researchers have started to focus on the semantic level of language. The objective of those research efforts is to learn the representation of language by extracting latent semantic relationships among the words. Section 3.2 examines the major developments in semantic-based methodologies used in modern NLP systems.

3.2 Bridging the gap between deep learning and NLP

Since the advent of deep learning, methods and approaches used to understand natural language have radically changed. Models using deep learning have proven highly effective for tasks related to NLP and its semantic understanding. Compared to traditional methods, they focused on the encoding of text into latent vector spaces to capture the relationship between text and its meaning through the understanding of the context.

3.2.1 Semantic word embeddings

Aware of the limitations of previous text vectorization approaches, leveraging the generalization capabilities of deep learning, the new generation of text embeddings was based on the *distributional hypothesis* [59], according to which, words that occur in similar contexts tend to have similar meanings. While LSA represented the first attempt to apply the distributional hypothesis in the field of NLP, it was not able to learn the semantic relationships among words. The advancement in computational power and the theoretical basis of the hypothesis enabled the design of machine learning models for mapping words into high-dimensional vector spaces so that similar words will be mapped to similar vectors.

The first approaches generate these dense vector representations, for individual words, using shallow neural networks. Word2Vec [123] and GloVe [142] are some examples of word embedding models. These approaches relies on *self-supervised learning* to automatically generate effective representations of words, without needing any manual annotation. Their training procedure was based on the definition of a context window C that defines the context associated with a given target word w as concatenation of the k previous and k next words in the text sequence. The Word2Vec model consists of a single fully-connected layer that can be trained using two different methodologies,

- *CBOW* (Continuous Bag of Words): The model takes as input the context C and is trained to predict the target word w.
- *Skip-Gram*: The model takes as input the target word w and is trained to predicts the words that are part of the context C.

A sketch of both training procedures is shown in Figure 3.1. The model map each unique word in the training corpus to a single vector. In the beginning, word vectors are randomly initialized and they are adjusted during the training process using *CBOW* or *Skip-gram* training objectives. It is worth noting that the training process does not need any human annotation and can run in unsupervised settings. Once trained, word embeddings shows the ability to capture the semantic relationships

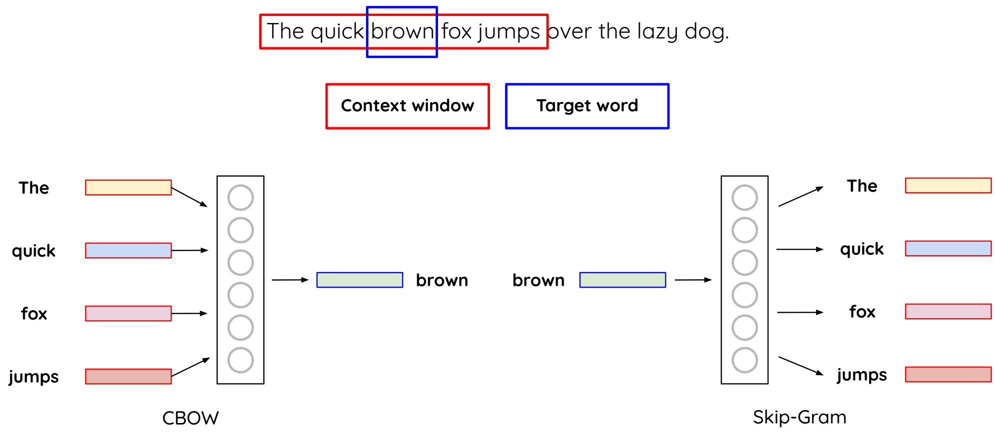


Fig. 3.1 Sketch of Word2Vec training strategies.

among words. The similarity between two words (w_i, w_j) is often computed using the cosine of the angle between their vector representations $(\vec{w_i}, \vec{w_j})$.

$$sim(w_i, w_j) = \frac{\vec{w_i} \cdot \vec{w_j}}{\|\vec{w_i}\| \|\vec{w_j}\|}$$

This similarity measure has become the de facto standard for measuring the degree of semantic similarity between two words. Word embedding models are used both in research and commercial NLP systems to overcome the problems of traditional methods, such as the bag-of-words approach. However, they still have limitations when it comes to semantic content understanding,

1. *Out-of-vocabulary words (OOV):* vector representations are generated only for words that appear in training data. In the case of a new word, they are unable to infer the vector representation.

2. *Contextualized representation:* each word is represented by a single static vector. During training, the context is used to learn the vector mapping, but once trained, the vector remains the same regardless of the context in which the word is found.

3. *Sentence and document representation:* text snippets are vectorized by averaging the vectors for the constituent words, losing sequential structure and context of the sentence.

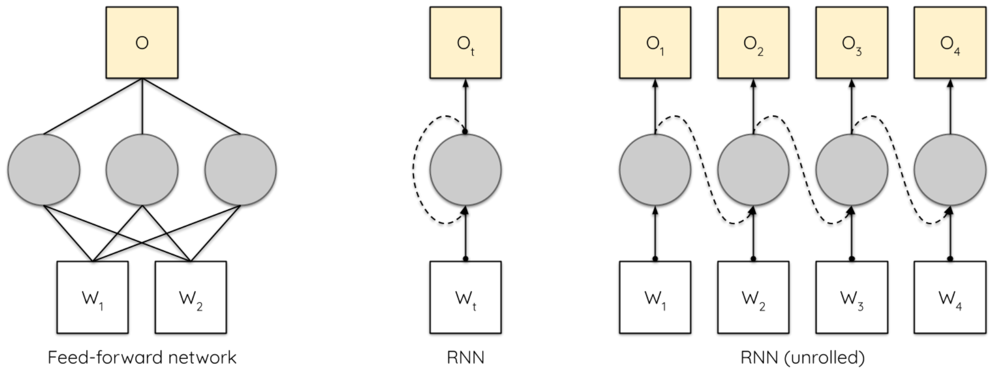


Fig. 3.2 Visualization of feed-forward and recurrent architectures.

The OOV issue has been partially mitigated by leveraging sub-word units and word compositionality [12]. The sub-words are generated using character n-grams, and allow for the extraction of vector representations for out-of-vocabulary words by adding the vectors of their constituent sub-words. However, limitations 2 and 3 remained unsolved. The static nature of word embeddings, as well as the aggregations required for document/sentence representations, adversely affect the semantic understanding of text.

3.2.2 Recurrent neural networks

Recurrent neural networks (RNNs) define a specific neural network architecture that can process sequential data. In their original definition, they extend standard feed-forward networks for handling sequences of arbitrary lengths [40]. Due to their inherently sequential nature, this design is particularly suitable for the NLP domain, where textual data should be analyzed within their context. This is possible through the introduction of cycles in network design that allows the information flow to persist through different time steps.

Compared with standard neural networks, RNNs have a feedback loop that allows them to exploit previous inputs during the computation at a given time step. Figure 3.2 illustrates the architectures of a feed-forward network and a recurrent network. Using the two variants on the right, we can compare the same RNN architecture in standard form and unrolled form for a sequence of length 4. In the unrolled form, for each time step, the RNN cell is replicated, allowing extended visualization of its inputs, outputs, and feedback loops. Recurrent architectures have been applied to a wide range of tasks such as machine translation [39], text summarization [129] and speech recognition [124]. The recurrent architecture is capable of handling long sequences, but suffers from vanishing gradients problem during training. LSTMs [62] have been shown to partially mitigate this problem by introducing a gate mechanism that permits better gradient flow through the network. Recurrent networks are capable of reaching state-of-the-art accuracy in a variety of domains but require long training on large corpora. Training recurrent networks is thus computationally expensive. The computation at each time step depends on all the previous steps, making parallelization of training algorithms more difficult.

3.2.3 The transformer model

The application of deep learning methodologies to NLP has led to several developments over the years. Nevertheless, large training datasets are also necessary to enhance model generalization and effectiveness. As a result, training such models has become much more time consuming and computationally intensive. The primary limitation of recurrent neural networks, discussed in Section 3.2.2, is their sequential structure. This hinders the parallelization of computation during training and inference, thereby slowing down the training process.

This problem was addressed by transformers [187], which have shown to achieve state-of-the-art results in a number of NLP tasks. The original transformer architecture is a sequence-to-sequence model proposed for the task of automatic machine translation. It is composed of an encoder and a decoder, the encoder is responsible for converting the source sequence into a sequence of vectors; the decoder is responsible for converting the vector sequence into the target text. Throughout the document, we will refer to the transformer as the encoder-decoder architecture. The most important novelty of transformer-based models is the introduction of attention mechanism that is in charge of learning the relationship between two elements in the sequence.

Attention mechanism The attention mechanism enables the model to focus on different parts of the input sequence by using learnable weights. Considering a sequence of n ordered tokens $S: w_1, w_2, ...w_n$, the transformer model represents each element w_i with a query q_i, a key k_i, and a value vector v_i. For a given token pair

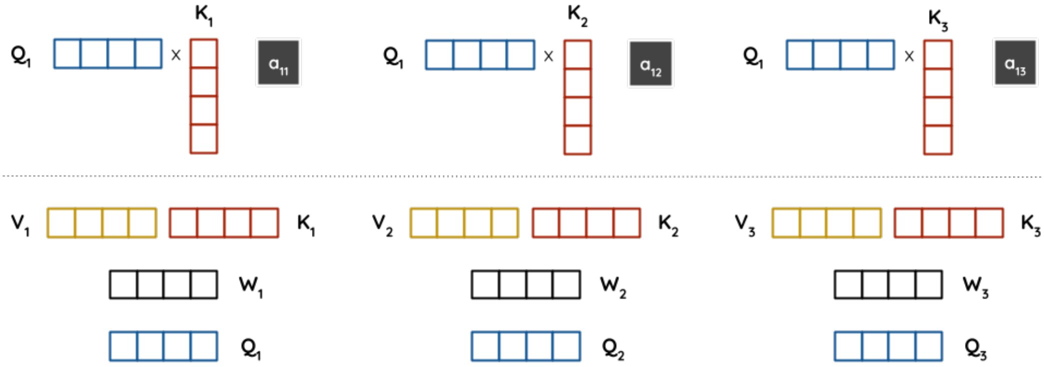
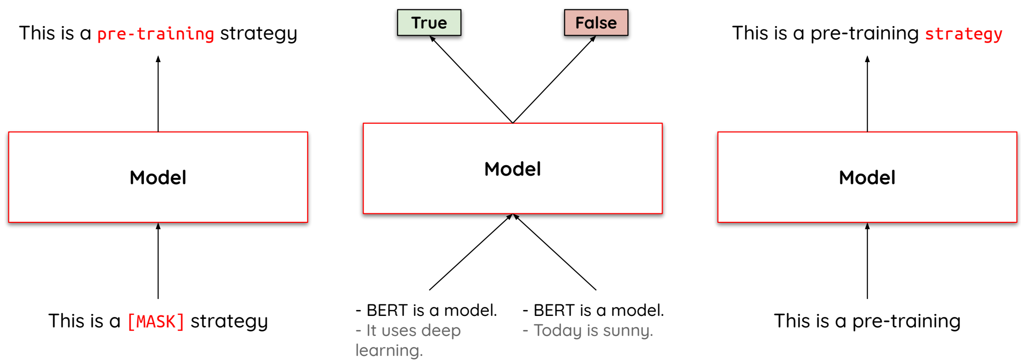


Fig. 3.3 Visualizing the computation of attention in transformer-based models.

(w_i, w_j) the self-attention score is computed as the dot products between the query vector and key vector. Figure 3.3 shows the inner working of attention mechanism for a sequence of three tokens. The illustration shows the first step of the computation, when attention scores between the first token and all subsequent ones are calculated.

The dot product between the query vector of the first word q_1 and all other key vectors is computed to obtain the attention scores a_{11}, a_{12}, a_{13} that refer to the relative importance of the first, second and third token, respectively, for the semantic representation of w_1. Note that, at each step, attention scores are also estimated for a given token and itself, thereby defining *self*-attention. To compute the context vector for a given token, the attention scores are normalized by applying the softmax function to obtain a probability distribution. Using normalized attention scores as weights, the context vector is obtained as a linear combination of the value vectors v_i of all tokens in the sequence.

Unlike recurrent networks, transformer models allow the computation of context vectors to be parallelized, improving the computational efficiency of the model. Self-attention, on the other hand, requires the computation of attention scores between each pair of tokens in the sequence, resulting in an $o(n^2)$ complexity with the sequence length n. Due to this limitation, the inputs and the outputs of modern transformer architectures are limited to a fixed threshold, usually 512 or 1024 tokens. Recent advances in transformer architectures have improved the computational efficiency of transformer models by introducing the concept of sparse self-attention [9], thus allowing the model to focus on a subset of the entire text and process longer sequences.



(a) Masked language modeling (b) Next sentence prediction (c) Language modeling

Fig. 3.4 Examples of different strategies for self-supervised learning.

3.2.4 Natural language understanding

Natural Language Understanding (NLU) is a subfield of NLP whose goal is to extract structured information from natural language. The original transformer model was proposed for sequence-to-sequence tasks: both the input and the output of the model are a sequence of tokens. NLU tasks, on the other hand, analyze the input sequence and provide a vector representation for each token in the sequence. This is exactly what the encoder in transformer models is trained to do.

BERT (Bidirectional Encoder Representations from Transformers) [38] is an encoder model based on the transformer architecture. It leverages a two step process to address NLU task: pre-training and fine-tuning. In the pre-training stage, the model learns to create an effective representation of the language by exploiting large corpora. This step does not require task-specific annotations, and it can be performed with only unlabeled data. In the second step, the pre-trained model is fine-tuned by leveraging task-specific annotations. During self-supervised training, the model leverages specific pre-training tasks,

- *Masked language modeling:* the model is presented with a text sequence containing masked words (i.e., each word is replaced with the standard token `[MASK]`) and its objective is to predict the original masked words. A sketch of the process is shown in Figure 3.4(a).
- *Next-sentence prediction:* the model is presented with pair of sentences. In one case, the second sentence follows the first in text and the others are randomly

selected. The model's objective is to predict whether the second sentence is consequential to the first one. The process is depicted in 3.4(b).

This approach allows the generation of training examples without explicit annotation and the model can learn a representation of language that is effective for many NLU tasks. The model is then fine-tuned by adding a specific head (e.g., classification layer) on top of the pre-trained model. Literature refers to this approach as *transfer learning*, since it transfers the knowledge of the pre-trained model to a specific task. Since the proposal of the original BERT model the idea behind encoder-based architectures, pre-trained using self-supervised strategies, has been extended by proposing dynamic masking [105], windowed attention patterns [9] and different pre-training objectives [27].

3.2.5 Natural language generation

Natural language generation (NLG) is the process of creating text with a specific meaning in natural language. Unlike NLU, NLG's primary objective is not to comprehend a text, but to generate new one with a certain meaning. The use of attention mechanisms in deep learning models has recently proven to be very successful for addressing the NLG tasks. In particular, it allows the model to focus on the most relevant parts of the input while generating the output. GPT (Generative Pre-trained Transformer) [149, 150, 13] identifies a series of decoder-based models that have shown great performance in various NLG tasks (i.e., narrative text and dialogue generation). They use the transformer's original decoding architecture and incorporate self-supervised learning for model pre-training. Considering the language generation objective, the pre-training strategy is formulated as a language modeling task: the model is trained to predict the next word in a sequence given the previous context. An illustration of the self-supervised strategy is shown in Figure 3.4(c).

Those models have shown impressive generalization capabilities and have outperformed many prior models on various NLG tasks. Recently, GPT-3 [13] has been proposed as one of the largest neural language models with 175 billion trainable parameters. It has demonstrated impressive ability to perform NLG tasks, including translation [58], question answering [128], and poem composition [36], even without model fine-tuning on those specific tasks.

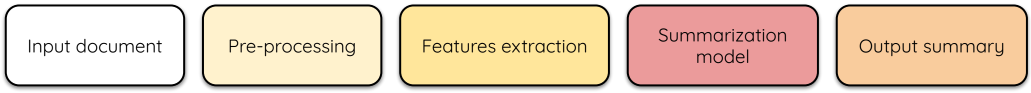


Fig. 3.5 High-level overview of the automatic summarization pipeline.

3.3 Automatic Text Summarization

Automatic summarization is one of the primary topics for most of the research works proposed in this thesis. This is one of the most studied tasks in NLP and, in its most general form, can be defined as the process of extracting or generating a short and coherent summary from a longer document. The original definition has been extended to cover different use cases, such as the summarization of multiple documents, structured data, or multimedia content.

3.3.1 General framework for automatic summarization

The task of summarizing a document can be considered a specific area of general document understanding; indeed, this can be viewed as a way to extract the most relevant and useful information from a document. Even though this task could be defined differently according to specific requirements and use cases, it is possible to define a general framework that describes the main steps involved in an automatic summarization system.

Figure 3.5 shows an high-level pipeline of the automatic summarization process. The first step of the process aims at defining the input of the system, which could be a single document or a collection of documents. This choice will have a direct impact on the next steps of the process, as the system will have to extract the most relevant information from the input data. According to the type of input data, the system can be classified as a *single document* or a *multi-document* summarization system. In the first case, it is assumed that the input is a single document and the system is responsible for extracting the most relevant information from it. Multi-document summarization systems, on the other hand, require multiple documents as input and they are responsible for combining information from several sources to generate a summary.

The second step of the process focuses on the *pre-processing* of the input data. This phase is aimed at cleaning, normalizing, and parsing the input data to obtain a structured format that can be processed by the system. At this stage, the input text is often split into sentences or words in order to be processed by the system. The pre-processing step may also include other operations, such as filtering, stemming, or stop-words removal. However, these operations are not always performed and they may vary according to the domain, the type of input data, and the requirements of the system.

Pre-processed data are then analyzed to extract a set of features that can be accessed by the summarization algorithm. The *features extraction* phase is aimed at gathering useful information that can be employed by the system to identify the most relevant information in the input data. This step is strongly dependent on the domain, the type of data, and the machine learning algorithm selected for the summarization task. Generally, it involves extracting traditional features such as the position or length of sentences in a document, but it can also employ more advanced features such as text embedding models that typically provide semantic-aware vector representations.

After the features extraction phase, the input data are fed to the *summarization model* that is responsible for generating the summary. It can be either supervised or unsupervised, according to the design of the system. In the former case, the model requires a set of training data that can be used to learn the parameters of the summarization algorithm. In the latter case, the system does not require a training set, and the summarization process is typically based on a heuristic approach. More details about different types of models are provided in the next sections.

Finally, the *output summary* is generated by the system. The summary is typically a shorter version of the original document and can be provided in various formats, such as a list of sentences or a set of key phrases. Depending on the summarization model, the output summary can be extractive (i.e., it is composed of sentences or fragments of the original document) or abstractive (i.e., it is composed of new sentences that are generated by the model and do not necessarily appear in the original document). A detailed description of both extractive and abstractive summarization models is reported in Section 3.3.2.

The general framework described in this section can be seen as a high-level pipeline describing the main steps involved in an automatic summarization system. According to the summarization models, data modalities, and different formulations of the problem, the pipeline could involve different steps and components.

3.3.2 Extractive and Abstractive summarization

As introduced in the previous section, automatic summarization approaches can be generally categorized into two different categories: *extractive* and *abstractive* methodologies. Extractive methods select the most relevant sentences in the input text to be included in the output summary. They do not require any phase of generation since the output summary is a combination of selected sentences. On the other hand, abstractive methods synthesize summaries by rewriting and paraphrasing the input text. Consequently, they enable the summarization process to go beyond simple sentence selection. Even though they both aim for the same goal, namely to create a summary that covers all relevant information of the text, they present different challenges.

Extractive summarization The idea of sentence selection underpins all extractive methods. The main challenge of this summarization task is how to select the most pertinent sentences to include in the summary. The majority of extractive methods determine the relevance of each sentence in the document first, and then rank them according to their estimated value. The summary is then composed by concatenating the most relevant sentences. A wide variety of extractive summarization approaches have been proposed in the literature, and they are heavily influenced by the characteristics of the input document. Chapter 4 provides an in-depth analysis at how extractive summarization is conducted on scientific documents.

Abstractive summarization Abstractive approaches generate a summary by writing novel sentences rather than extracting them. It is a more challenging task because, in addition to identifying the most relevant information, the summarizer requires the ability to compress and paraphrase the document content. Since the introduction of transformer-based models for abstractive summarization task, there has been a significant advance in this field; sequence-to-sequence models [92, 200] have shown impressive capabilities both in understanding and generation tasks. Transformer-based models, however, are computationally expensive and have difficulty handling long sequences (such as long documents). This hinders their practical use for the summarization of long texts and they require specific countermeasures to handle this limitation.

3.3.3 Unsupervised text summarization

Both the estimation of text relevance for sentence selection and the ability to write human-like summaries are challenging tasks, and often require supervision. In the summarization setting, supervised learning requires annotated data consisting of pairs of texts and their corresponding human-written summaries. Since summary generation is a complex task that cannot be easily automated, such data collections can be expensive and difficult to create. To overcome the lack of training data, unsupervised summarization approaches have been proposed for automatically extracting key information from texts. Using semantic similarity or lexical co-occurrence, these methods are able to identify key sentences in the input text. Due to the inherent complexity of generative tasks, unsupervised approaches usually rely on extractive summarization, i.e., selecting key sentences from the input text and concatenating them to create the final summary.

Among the most successful unsupervised methods there are,

- *Graph-based approaches* [122, 41, 152], which represent sentences in a text as nodes on a graph, with edges weighted based on a similarity measure between them. The relevance of a sentence is then calculated using a graph-based ranking algorithm, such as PageRank [138], to select the top-ranked sentences.
- *Topic modeling techniques* [114, 15] which comprehend a set of methods leveraging latent semantic analysis [90] to identify the topics in a text and then summarize its content by selecting the most relevant sentences for each topic.
- *Optimization-based models* [99, 98] that consider the text summarization problem as the maximization of a submodular function with a budget constraint.
- *Clustering-based methods* [48, 52, 89] that cluster sentences in the input text based on a specifically-designed distance metric and then pick the most representative sentences in each cluster for the final summary.

The community has focused on extractive approaches because they do not require text generation capabilities, which are difficult to automate. Yet, such approaches are less flexible than abstractive ones as they can only be used to select key sentences from the input text.

By leveraging the generation capabilities of large-scale language models [13] and their ability to capture long-term semantic dependencies among words, it has become possible to automatically *generate* summaries from texts. Those models usually leverages inductive biases in source document structure to generate training examples without explicit human annotation [206]. They also add specific pre-training objectives to enhance summary fluency and domain specialization [196]. However, when using abstractive summarization models, especially in unsupervised setting, it is not ensured that the generated summaries are factually correct. Often, summary sentences are distorted by hallucinations, making it difficult to apply them to practical applications [115].

3.3.4 Learning to summarize text through human references

In several scenarios, unsupervised summarization models are not able to capture the essential information in the input text. In this setting, humans can provide reference summaries for training supervised summarization systems. Manually-written references can be used to train supervised models to generate summaries that are similar to the given examples. Training data for machine learning models usually consists of pairs of input texts and corresponding human-written summaries. While supervised approaches are often more successful than unsupervised methods, it remains difficult to produce summaries that are both accurate and fluent. Both extractive and abstractive summarization models have been proposed for supervised summarization.

Extractive strategies leverage sentence-level features to learn a machine learning model that is able to efficiently identify important sentences from the input text. The advancement in semantic text representation guided by deep learning models has fostered the design of advanced summarizers. The sentence selection and ranking has been implemented by using simple RNN-based classifiers that extract relevant sentences [129] or more complex training procedures leveraging reinforcement learning [130]. Contextualized text representation is extensively used in those models to estimate sentence scores. Some models propose to produce more effective sequence representations by designing hierarchical architectures to represent sentences [205] or even the whole document [203]. The introduction of transformer-based models, which learn text representations leveraging attention mechanisms, has led to the proposal of extractive summarization methods, which estimate sentence relevance by exploiting inconsistencies among attention scores at different encoding levels [66].

The abstractive model, on the other hand, can generate text summaries that match human-written references. The literature on abstractive summarization includes a variety of methods including template-based methodologies [137, 189] that rely on custom sentence structures to generate summaries, and graph-based approaches [126, 176] that build a graph data structure to better represent the document. More recent approaches use comprehensive neural models rather than traditional approaches that are composed of individual components [100]. They rely on sequence-to-sequence architectures that use an encoder to process the input text and a decoder to generate the abstractive summary. The literature includes a wide variety of models leveraging convolutional [158] or recurrent [141] networks for encoding the input text and recurrent networks for the generating the summary [77].

In 2018, the introduction of the Transformer architecture [187] created a significant impact on the field of abstractive summarization. Custom pre-training strategies have been shown to be effective for training abstractive summarization models on un-annotated corpora [92]. The most effective pre-training strategies exploit (i) multi-task learning to generate more effective text representations [151], (ii) the generation of important sentences that have been masked from the source document [200] and, (iii) the joint optimization of multiple word prediction using prior context [145].

Lately, the use of large transformer models in reinforcement learning setting achieved impressive results for abstractive summarization [172]. Large language models are trained on explicitly collected human feedbacks and effectively generalize to unseen contexts and domains. Despite the clear advantages of this approach, it has many disadvantages, including the need for a large number of expensive human feedbacks. Furthermore, training large models can be extremely time-consuming and expensive, limiting the reproducibility of these methods.

3.3.5 Metrics for evaluating text summarization approaches

The evaluation of automatic summarization systems, or, more broadly, language generation models is a difficult task that has been extensively studied in the literature [160]. The ability to compare results between different methods is essential to measure the progress of any scientific field. In text summarization, the evaluation is typically performed by comparing the generated summaries with the references annotated by humans. A good candidate summary should have the following properties:

- *Informativeness*: the summary should contain the most important aspects of the source text.
- *Fluency*: there should be an appropriate syntactic structure and proper grammatical relations between sentences.
- *Conciseness*: the summary should be as succinct as possible while retaining the main ideas of the source text.

Currently, there is no single solution for the automatic evaluation of summarization systems that can effectively assess all these properties. At present, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [97] is the most common evaluation framework. It comprehends a set of scores that measure the overlap between system-proposed summaries and its corresponding human reference,

- *ROUGE-N*: it measures the n-gram overlap between the predicted and reference summary. A n-gram is defined as a sequence of n consecutive words. Typical values for n used in the literature are $n = 1, 2, 4$. They are typically used to measure the *informativeness* of the system summary.
- *ROUGE-L*: it measures the longest common subsequence (LCS) between the predicted and reference summaries. LCS is a sequence of words that appear both in the predicted and reference summaries.
- *ROUGE-W*: it is a weighted variant of ROUGE-L score. Summaries that contain several common *consecutive* words are preferred over those with only longer common sequences. Keeping track of the number of consecutive n-gram matches, it rewards more sentences with higher numbers of consecutive

matches. Together with ROUGE-L they are commonly used to assess the *fluency* of the proposed summary.

- *ROUGE-S*: it measures the bi-gram (i.e., n-gram with $n = 2$) match between the two summaries. In this case, bi-gram are not only consecutive words but allow gaps of any length.

For each score, some settings (e.g., when summaries have fixed length thresholds) consider recall as reference metric, while in other cases, also precision and F1 measures are included in the evaluation. ROUGE-scores have shown to be highly correlated with human judgement, thus they are the de facto standard for the evaluation of text summarization systems. However, they also have some relevant limitations. First, they only consider the lexical similarity between the predicted and reference summary. Second, they only consider whether a word (or a sequence of words) appears in the summary, and disregard its context or position. ROUGE-scores were originally proposed for extractive summarization, in which the reference summary is often a distilled version of the source document.

Since the community has made remarkable progress in abstractive summarization, mostly due to neural models, it has begun to consider more effective evaluation metrics that aim to overcome some of the limitations associated with ROUGE-scores. Especially in the context of generative tasks, metrics need to extend beyond lexical similarity and consider also the semantic relationship with the reference summary. Recently, the community has proposed a number of alternative evaluation metrics to address the limitation above.

- *BERT-score* [202] is a metric that leverages the similarity between tokens in a candidate text and their counterparts in the reference text. It leverages token representations obtained by pretrained transformers models. Token similarity is obtained by computing the cosine similarity between the contextual word embeddings.

- *Sentence-BERT score* [153] is computed by exploiting transformer-based architectures specifically trained for the Semantic Textual Similarity (STS) task. Unlike BERT-score, it computes semantic similarity comparing the representations of entire reference and candidate summaries.

- *SUPERT* [49] is a metric that does not require the use of any reference summary. It was originally proposed for the multi-document summarization task (i.e., provide a single summary for multiple source documents). It creates pseudo-reference summaries automatically and then leverages specific transformer-based models to compute the semantic similarity with the system-proposed summary.

Although the community has made progress in this direction, it is still seeking alternative solutions to better capture the semantic similarity to the reference summary [43]. The above list is by no means exhaustive, but rather it is meant to provide an overview of the current state-of-the-art.

Even though researchers have made significant progress toward the development of automated summary assessment, the gold standard remains *human judgment*. Unfortunately, the process is time-consuming and expensive, making it unsuitable for large-scale summarization. Also, human evaluation is usually not reproducible and highly subjective, and as a result, is unreliable as the only metric for algorithm development or benchmarking.

Chapter 4

Natural Language Understanding for Scientometrics

Scientometrics is the research branch that applies statistics and computational methods to the study of scientific literature. Its major goal is to quantify and visualize patterns and trends in scientific publications. In scientometrics, NLP can be applied to extract and analyze large amounts of bibliographic data from a quantitative perspective. This chapter provides a review of the related literature in Section 4.1 and introduces the research contributions in the analysis of scientific citations in Section 4.2. We then provide the details of methodologies for the extraction of results-oriented highlights (Section 4.3), facet-specific summaries (Section 4.4), and unsupervised extraction of presentation slides (Section 4.5). Finally, Section 4.6 introduces some of the current challenges and provides an overview of the future directions in the domain of scientometrics and scientific summarization research. A graphical taxonomy of the contributions featured in this chapter is provided in Figure 4.1.

4.1 Natural Language Processing for scientometrics

NLP methodologies can be used to address several tasks in scientific literature analysis. The contributions discussed in this chapter focus on the analysis of bibliographic data. The objective is to extract structured information needed to support scientific research and knowledge discovery.

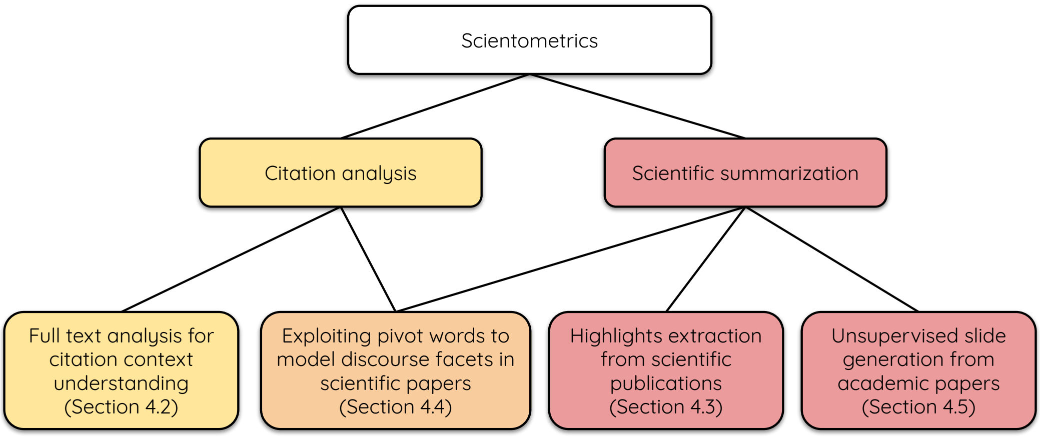


Fig. 4.1 Graphical overview of the topics and contributions covered in Chapter 4

An essential aspect of scientometrics is the analysis of citation context (i.e., the text surrounding the citation) to aid in its semantic understanding [72]. Section 4.1.1 discusses research works that exploit the citation context to automatically extract information about the citation purpose.

Scientometrics literature, however, also includes an high number of studies aiming at summarizing the content of scientific articles. It involves the development of automated summarization techniques tailored to address challenges in the scientific domain. These techniques are discussed in Section 4.1.2.

4.1.1 Citation analysis

Citations are an essential part of scientific literature. They are used to (i) reference other authors' previous work, (ii) recognize the influence of prior work on the current research, and (iii) support the scientific claims of a study. Therefore, understanding the semantics of citations is of paramount importance in the scientific domain. Citation context analysis helps in this understanding by extracting information from the text surrounding the citation.

Large-scale data collections The use of machine learning algorithms in this domain is advancing at a rapid pace due to the availability of large datasets [73, 29, 197, 22].

SciCite [29] is one of the most comprehensive data collection containing annotations for the citation intents of a variety of scientific publications. Each reference is annotated with one of the following classes:

- *Background*: when the citations provide context, which includes details about a topic or problem. They are also used to highlight some specific points within the publication.
- *Method*: when the citations mention some specific methods, tools or datasets which are used in the current study.
- *Results/Comparison*: when the citations are used to support or compare results presented in the current study.

The dataset covers 6,627 publications containing 11,020 annotated references.

ACL-ARC [73] is a smaller, yet large-scale dataset containing annotated publication in the field of *Computational Linguistics*. It covers a more fine-grained set of classes that comprehends: *background*, *motivation*, *uses*, *extension*, *comparison/contrast* and *future works*. The complete dataset contains 186 paper with a total of 1,941 annotated references.

Following a similar approach, a more comprehensive dataset has been proposed by Pride and Knoth [144]. It is built on the premise that the contributors of the papers included in the data collection can provide the most accurate annotation for each citation. It contains 11,233 references annotated by surveying the original authors of each publication. At present, this is the most comprehensive dataset in terms of variety of annotators and number of disciplines.

CL-SciSumm [22] is a research competition aiming at fostering academic research in the automatic summarization of scientific articles. The organizers propose the identification of the citation intent as one of the tasks. Given a collection of publications P and a citing paper $c \in P$ that cites a referenced paper $r \in P$, the proposed task is to identify the span in r that mostly reflects the citation in c. In addition, for each identified citation span, the organizers propose to classify what facet this citation is focusing on. To address the aforementioned tasks the authors propose the manual annotation of a portion of SciSummNet dataset [197]. The annotation covers the references of 40 publications belonging to the field of *Computational Linguistics*.

In early works on citation context analysis, a linguistic approach was proposed. Citation context has been used as an indexing tool for retrieving citations [121], to identify its polarity and potential impact [61], or to build classification models that can extract additional information [177, 167]. In recent years, large-scale benchmarks have allowed researchers to identify the intent behind a specific citation using data-driven methodologies [70]. Several studies have explored the use of both traditional and contextualized word embedding to automatically determine citation intent [157, 74]. The semantic analysis of citation context has also been applied to extend beyond the detection of the citation intent and identify specific roles or properties in the citation context [44].

Even though NLP tools have shown to be extremely effective at studying citation context, citation networks can provide additional information that can be helpful to understand the dynamics of the citation process. Using graph modeling [10] or multi-task learning [29], citation networks have been used to enrich citation context with additional information that can contribute to the intent detection task. The development of semantically-linked publication networks can also be supported by integrating structured data from the citation network and high-level information from the citation context [45].

Many other research directions have arisen from the broader scope of citation context analysis. It has been used to develop large scale citation indexes [135], detect citation polarity [69, 191], and categorize publications based on referenced articles [120].

4.1.2 Scientific document summarization

The automatic summarization of scientific publications involves composing brief summaries of research articles. In a nutshell, the objective is to provide readers with an overview of the paper so that they can quickly understand the most relevant aspects. Clearly, this task has several practical applications. Researchers are often overloaded with research material available on their field and spend considerable time browsing full-text papers and abstracts. Even though the abstract can be considered a summary itself, it is generally provided by the authors and may not contain all of the necessary information to fully understand the purpose, approach and contribution of the research. In contrast to short-document summarization, scientific summarization involves several unique aspects,

- The structure of scientific papers tends to be well-defined. It usually includes an introduction, sections describing the methodology, results and discussion [168].
- Using citation networks and sharing common ideas, scientific papers can benefit from external sources for summarization [197].
- The length of scientific papers is generally long and they usually contain lots of information. Consequently, the summarization task may become more challenging.
- The full text of the article may have mathematical symbols and equations, depending on its field of research. Additionally, the terminology can be specific to a field and may require specialized knowledge. This can interfere with document understanding and hinder the automatic summarization process.

These unique features of scientific papers have stimulated extensive research in this area. Much of the current literature on scientific summarization pays particular attention to extractive methodologies. Researchers have proposed machine learning models trained on specific sets of features to extract general-purpose summaries [83, 94]. Scientists, however, look for several perspectives and aspects of a paper in the summary, since scientific publications are typically multifaceted. The analysis of paper citations has been shown to be helpful both for general-purpose summarization [119] and for generating facet-aware summaries [31, 32]. Supervised learning can be also used to tailor the summarization process for specific objectives, such as the extraction of results-oriented highlights that help the reader grasp the key findings of a scientific publication [33].

Although it can be a more interesting problem from a research perspective, the field of abstractive summarization has received less attention due to its inherent difficulties. However, the recent advances in neural models has helped to bridge the gap between extractive and abstractive summarization. For the generation of facets-specific summaries, recurrent networks have been used. These networks effectively comprehend context and paraphrase peculiar aspects of the source document [30]. Recent trends in text generation have been driven by transformer-based architectures.

The problem, however, is that these models cannot effectively handle long sequences, preventing them from processing long documents.

To overcome this limitation, they are mostly used for the extreme summarization task (e.g., to analyze a shorter version of the document to produce a single sentence summary) [14, 110] or combined with extractive approaches (e.g., abstractive summarization is applied on a selection of relevant sentences). Recently, transformer models have been extended to handle long sequences by introducing alternatives to the original attention to prevent exponential complexity growth with sequence length [9, 78]. In the context of scientific abstractive summarization, these models have shown to outperforms hybrid extractive-abstractive methodologies generating short and coherent summaries for scientific documents [194].

4.2 Full text analysis for citation context understanding

The analysis of citation context has shown to be useful in several scenarios. Our contribution in this field aims at analyzing the value of the full text of the paper for understanding the citation context [85]. Paper's sections can be usually divided in two main classes: open and closed access. Open access sections (i.e., title and abstract) of articles are publicly available, while those in closed access are usually available to institutional partners or by paying a fee. The full text of a paper contains primarily closed access sections that provide the most detailed and accurate description of the study's findings.

Considering a paper's incoming citations and their citation context (i.e., citances), in our study, we seek to identify those sections whose content is relevant to the citation context, thus, reading them is recommended. Based on the categorization of open and closed access sections, our goal is to answer the following research questions:

1. *How can we analyze the semantic correlation between citations and section in the reference paper?*
2. *Can we use machine learning to identify citations that need full-text exploration, beyond just reading the abstract and title of cited articles?*

To answer the aforesaid questions we leverage the ScisummNet dataset [197]. It contains 1,000 papers in the field of *Computational Linguistics* with their corresponding abstract, full text and citation networks (e.g. citation sentences, citation counts). Each citance is labeled with the sentences from the cited paper that are relevant to the citation context. This is the only dataset containing fine-grained labels for citances, thus providing an unique opportunity for measuring the correlation between citances and the full text of the reference paper.

4.2.1 Correlation analysis

To estimate the correlation between citances and each section of the cited paper, we build a classification model that can predict whether a given citance points to a specific publication. The machine learning classifier leverages a set of features identified by calculating similarity scores between the citation and each section of the cited paper,

- *Semantic similarity:* we estimate semantic similarity using word- and sentence-level embedding models. For this purpose, we use the Word2Vec [123] model for word-level similarity and, Sent2Vec [139] and BERT [38] for sentence-level comparisons.
- *Syntactic similarity:* we measure lexical similarity using ROUGE-1, -2, and -L scores [97] to compare the *n*-grams overlap at different levels of granularity.

The features are used to generate a binary classification dataset, where each record is denoted by the pair (p, c), where p represents a paper and c represents a citance. Records are assigned a positive label if c references p, and a negative label otherwise. We use the dataset to train classification models that can estimate the significance of input features in the classification process. This will enable us to gain some insights related to the first research question. To ensure consistency across different papers, we apply domain-specific regular expressions to section titles and categorize them into a pre-defined set of classes: *title*, *abstract*, *introduction*, *related works*, *method*, *experiments* and *conclusions*.

The classification models include Decision Trees, Random Forests, Gradient Boosting, and AdaBoost, which allow the extraction of feature importance. Figures 4.2 and 4.3 show the importance of sections and features type, respectively.

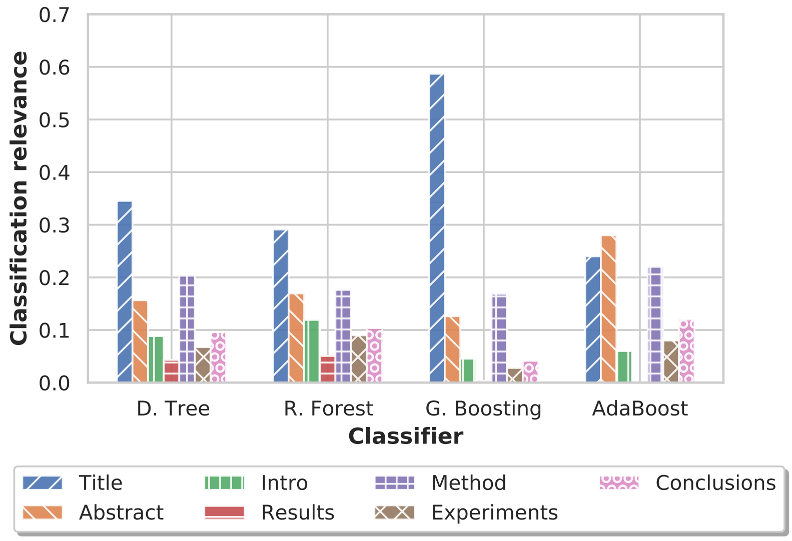


Fig. 4.2 Analysis of features relevance, separately per section

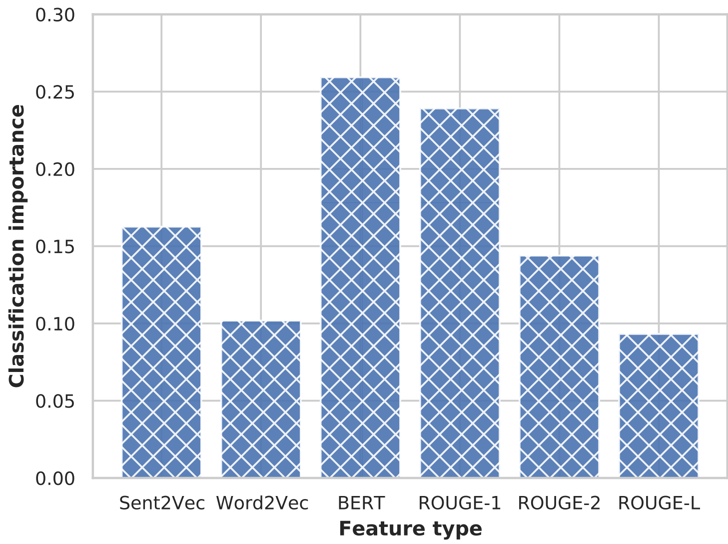


Fig. 4.3 Analysis of features relevance, separately per type

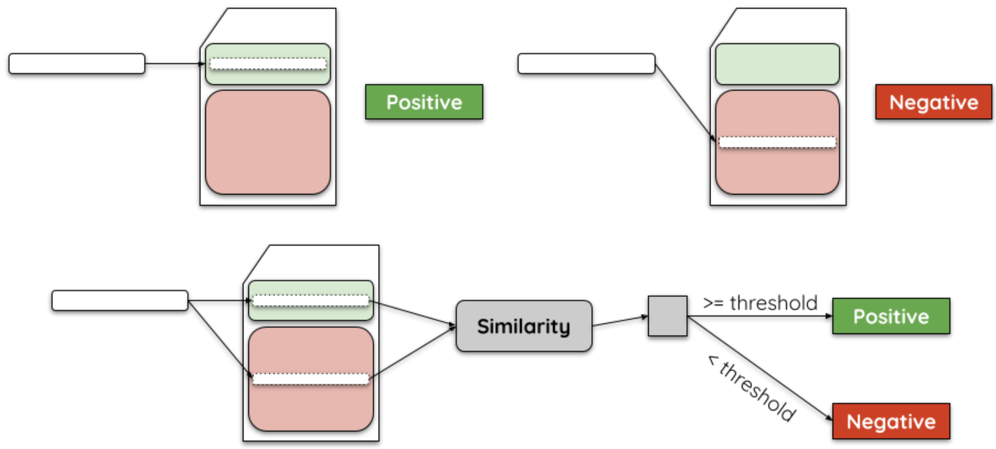


Fig. 4.4 Dataset generation procedure, citance classification.

Title, *abstract*, and *method* are clearly the most influential sections overall. The strong correlation with open access sections supports the investigation for our second research question. Among the features types considered in the experiment, BERT-based similarity is selected as the most relevant. Most likely, this is due to the semantic information that BERT has the capability of capturing, which other feature types do not. The second most relevant feature type is the ROUGE-1 score, which indicates that syntactic matches between individual words can have an impact on classifier predictions.

4.2.2 Identification of citation contexts requiring full-text reading

The second research question is about automating the identification of cases when reading the open access sections of the paper is not enough to understand the citation context. In other words, it aims at discovering when reading the full text of the paper is required to understand a citation from a referencing paper. It could support researchers by identifying those instances in which they ought to read a paper's full text to gain a complete understanding of the citing context.

To build the classification models we leverage the annotations provided in the ScisummNet dataset. Each citance c could refer to one or multiple sentences of the reference paper $s_i \in r$, found in either the open or closed access sections. We create a binary classification dataset by labeling as positive the citances referring to open access sections, as negative the ones that refer to closed access sections; if a

Table 4.1 Performance analysis of the classifiers on the ScisummNet dataset. $\alpha = 0.5$

Classifier	**AUC**	**Accuracy**	**Negative class**		
			Precision	**Recall**	**F1-Score**
AdaBoost	0.90	0.90	0.92	0.97	0.94
Gradient Boosting	0.91	0.91	0.94	0.96	0.95
Decision Tree	0.71	0.87	0.93	0.93	0.93
Random Forest	0.90	0.91	0.93	0.97	0.95

citance refers to both open and closed access sections, we apply a similarity-based labeling mechanism. When the similarity between the referenced sentences exceeds a threshold α, we label the citance as positive, otherwise negative. Figure 4.4 depicts the procedure for citance labeling. Each entry in the dataset is identified by pair (p, c) where p is the reference paper and c is the citance referring to p. Several threshold values are evaluated without significant differences in the classification results [85].

We evaluate the classifiers' performance using the Area Under Curve (AUC), overall accuracy, and precision, recall and F1-score of the negative class. To evaluate whether the classifier can successfully address the most significant cases, we focus on the results of the negative class (i.e., the classifier predicts that the full text analysis is required). Results reported in Table 4.1 shows that all the classifiers achieve high accuracy and F1-score, with a lower performance for Decision Tree. Both the precision and recall of the negative class are higher than 90%, thus indicating that the classifier is able to successfully identify the cases that require the full text analysis.

An article's full-text certainly provides additional information that can be helpful to the reader. However, it is not clear if exploring sections of the paper beyond the title and abstract will yield better insights into citing snippets. According to the findings of this study, lexical and semantic features can be used to train a classification model that differentiates between cases that show clear benefits and those that do not. The classifier is able to accurately predict the majority of the cases where the full text is not expected to provide additional insights into the citing snippets. The results of this analysis demonstrate that this approach can effectively help in reducing the effort required for citation analysis, which can be a time-consuming process.

4.3 Highlights extraction from scientific publications

The summarization of scientific publications has received significant attention in the field of scientometrics. Summarization models for scientific publications aim to provide (i) a concise overview of the most relevant aspects of the documents or (ii) a summary that highlights specific content according to the user specifications.

Reviewing related literature is usually a demanding task for scholars, requiring them to browse through an extensive collection of documents. Usually, they focus on identifying the main findings of the publications to determine if they apply to their research. To facilitate this task, several research journals have developed *publication highlights*, brief summaries of findings, in bullet-point format, that deliver a quick overview of the most significant results reported in the research paper[1]. These summaries are usually authored by the paper's authors and are usually limited to 3 to 5 sentences.

Our contribution in this field aims at providing a novel method to automatically extract paper's highlights from the full text [16]. To address this problem, we design a supervised summarization pipeline tailored to the selection of highlights. The goal of the proposed method is to facilitate annotation of newly published articles (by providing suggestions for highlights) as well as automatic extraction of highlights from past articles that do not include them, which is a common problem in older publications. Our model relies on regression algorithms to identify K sentences within a scientific article, whose content is most likely to be correlated with its highlights (K is an user-defined parameter, generally set between 3 and 5).

4.3.1 Feature extraction

To analyze article content, full-text articles are divided into sentences using punctuation, and each article $A_i \in \mathscr{A}$ is represented as a set of distinct sentences $\{s_1, s_2, \ldots s_n\} \in A_i$. To represent sentences, we define a specific set of features.

- *Symbols count*: it indicates the number of non-alphabetical symbols in the sentence. Symbols will be used only rarely in highlight sentences because they

[1] https://www.elsevier.com/authors/tools-and-resources/highlights

are designed to provide a brief summary of the article's findings rather then specific details of the methodology.

- *Parts Of Speech (POS)*: It is a set of features indicating the presence of nouns, adjectives, conjunctions, proper nouns, and present-form verbs in the sentence. According to Krapvin et al. [80], there is a strong correlation between the presence of specific POS types in the sentence and its relevance.

- *Sentence position*: it is a feature indicating the position of the sentence in the document. We design a position score ranging between 0 and 1. It is computed according to the following formula:

$$f(s) = \begin{cases} -\dfrac{l_s}{m/2} + 1, & \text{if } l_s \geq \dfrac{m}{2} \\ \\ +\dfrac{l_s}{m/2} - 1, & \text{if } l_s < \dfrac{m}{2} \end{cases} \tag{4.1}$$

  Where l_s is the sentence offset with respect to the initial sentence of the paper, and m is the maximum offset in the considered article. This feature is designed to reward sentences that appear at the beginning or at the end of the paper since these parts usually contain the most informational content, while the center of the paper generally includes more technical details.

- *TF-ISF word relevance [131]:* it corresponds to the Term Frequency-Inverse Sentence Frequency (TF-ISF), a feature that estimate the overall importance of a sentence. Those text passages containing many highly relevant terms tend to be correlated with highlights. To avoid biasing the TF-ISF score, we remove stopwords (e.g., very common words that does not convey any specific meaning) from the paper's sentences.

- *Semantic relevance:* we compute the overall semantic relevance of the sentence in the document by leveraging Word2Vec [123] and Sent2Vec [139]. This is done by modeling each sentence in the document as a node on a graph whose edges are weighted in accordance with the cosine similarity between sentence embeddings. For Word2Vec model, sentence embeddings are obtained by averaging the corresponding word vectors, while Sent2Vec already provides a single embedding vector for each sentence. PageRank [138] is used to estimate overall sentence relevance.

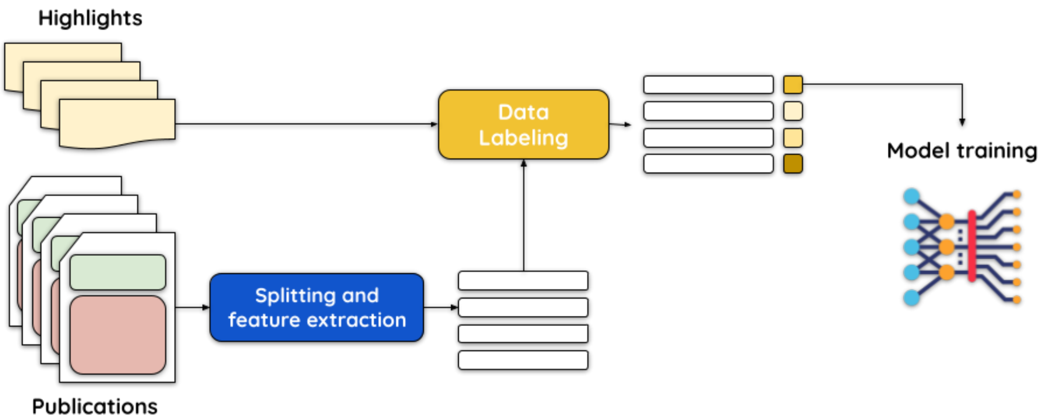


Fig. 4.5 Highlights extraction pipeline.

- *Sentence similarity with the abstract:* even though highlights and abstracts serve different purposes, the abstract of the paper usually contains the overall summary of the research paper. For each sentence we compute syntactic similarity (i.e., using ROUGE-L) and semantic similarity (i.e., using Word2Vec and Sent2Vec sentence representations) with the abstract of the corresponding paper.

These features are designed to capture the main elements that characterize a highlight sentence. For each sentence $s_i \in A_i$, we compute a feature vector $v(s_i) \in \mathscr{R}^d$, where d is the number of features defined above.

Regression model We train a regression model to estimate the highlight score of a given sentence $\hat{h}(s_i) \in \mathscr{R}$ in the range $[0,1]$. The advantage of regression models is that they predict continuous scores for the input sentences, allowing us to rank and select the top-K sentences as highlights. To obtain the highlight score of each sentence in the training set, it is annotated according to the maximum similarity score between the corresponding sentence and any of the article's highlights. We use ROUGE-L F1-score to estimate the similarity between a sentence and its highlights. In most cases, papers have more than one highlight sentence, so the label score of candidate sentences is determined according to the maximum score with the highlights. The extraction process does not take into account sentences in open access sections of the paper (i.e., title and abstract), because highlights aim at providing additional information beyond those sections. Figure 4.5 shows the pre-processing and labeling steps to generate the data for the regression model.

4.3.2 Benchmark data collections

Table 4.2 Statistics for the CSPubSum, BIOPubSumm and AIPubSumm data collections.

		#	Paper		Abstract	
			#W	#S	#W	#S
CS	train	10131	8236.63	262.26	316.81	10.61
	test	150	6010.78	196.08	297.91	10.22
BIO	train	8068	4894.24	160.75	371.04	13.23
	test	2690	4946.15	160.91	364.54	13.16
AI	train	198	10594.16	344.73	429.73	13.43
	test	66	11028.37	352.89	413.66	13.36

We train and evaluate our approach on three different data collections specifically designed for automatic highlight extraction:

- *CSPubSumm* [33] consists of 10,131 and 150 articles for the training and test set, respectively. The articles are annotated with their corresponding set of highlights. It is a collection of journal articles from the Computer Science domains.
- *BIOPubSumm* [16] is our newly collected dataset containing articles from *Biology and Medicine* domain. It contains 10,758 articles split into into training and test set with a 80%/20% ratio, respectively. Each article is accompanied by a gold set of highlights.
- *AIPubSumm* [16] is a smaller data collection we collected from *Artificial Intelligence* domain. It includes 198 training articles and 66 test articles with their corresponding set of highlights.

Each dataset contains articles from different publications and they aim to cover a broad range of topics and writing styles. The statistics of the datasets are summarized in Table 4.2. #W and #S represent the average number of words and sentences in an article or in its abstract, respectively, while # represents the number of articles in

Table 4.3 ROUGE-L F1-score results comparison. Highest scores are reported in boldface and significant differences with respect to the best performing method are marked with * (p-value < 0.05).

K	Sub-Modular	Text-Rank	Lex-Rank	DT CLF	RF CLF	MLP CLF	GB CLF	LSTM CLF	DT REG	RF REG	MLP REG	GB REG
						CSPubSumm						
3	0.235*	0.209*	0.257*	0.276*	0.298	0.272*	0.273*	0.295	0.303*	0.313	0.309*	**0.316**
4	0.228*	0.205*	0.237*	0.258*	0.284*	0.254*	0.254*	0.278*	0.291*	0.297*	0.297*	**0.303**
5	0.213*	0.193*	0.217*	0.239*	0.265*	0.239*	0.240*	0.256*	0.270*	0.278*	0.278*	**0.284**
						BioPubSumm						
3	0.221*	0.208*	0.227*	0.248*	0.253*	0.250*	0.250*	0.243*	0.259*	0.275*	0.278	**0.28**
4	0.215*	0.197*	0.223*	0.236*	0.241*	0.239*	0.238*	0.231*	0.250*	0.265*	0.27	**0.271**
5	0.204*	0.185*	0.199*	0.222*	0.227*	0.225*	0.224*	0.219*	0.237*	0.249*	**0.258**	0.257
						AIPubSumm						
3	0.180*	0.195*	0.225*	0.256	0.277	0.27	0.268	0.235*	0.256	0.283	0.28	**0.289**
4	0.175*	0.187*	0.212*	0.247	0.271	0.252	0.253	0.226*	0.256	0.274	0.267	**0.281**
5	0.166*	0.177	0.201*	0.227*	0.263	0.235	0.236	0.215*	0.244	0.263	0.259	**0.266**

each data collection. The instructions to reproduce the data collections together with qualitative examples of the system output are publicly available[2].

4.3.3 Experimental results

In our empirical evaluation we evaluate several regression models that aim to estimate the score of each sentence in an article. Specifically, we evaluate Decision Trees (DT), Random Forest (RF), Gradient Boosting (GB) and Multi-Layer Perceptron (MLP) regression models. Using the same algorithms, we also compare our regression models with classification-based approaches and present the results for the three data collections. For a comprehensive evaluation, we also compare our approach to several unsupervised baselines (e.g., CoreRank [178], LSARank [171], Submodular [99], TextRank [122] and LexRank [41]) that estimate overall sentence relevance but are not tailored to highlights extraction.

We use the ROUGE score [97] to evaluate the quality of the automatically extracted highlights. Specifically, we use ROUGE-L F1-score to compare the quality of the candidate highlights with the ground-truth. Table 4.3 presents the results of the evaluation with *CSPubSumm*, *BioPubSumm* and *AIPubSumm* data collections. We report the results for 3 values of K that are commonly required by journal editors when preparing the paper submission (i.e., $K = 3$, 4 and 5). The column labeled as

[2] https://github.com/MorenoLaQuatra/domain-specific-academic-dataset

Sub-Modular, *Text-Rank* and *Lex-Rank* reports the performance of the unsupervised baselines, while the columns with *CLF* and *REG* refer to classification-based and regression-based models, respectively.

The results show that our regression-based models outperform unsupervised baselines and classification-based models across all data collections. Our best performing model is the regression model leveraging Gradient Boosting algorithm that achieves the highest average F1-score across all three datasets. The results show that regression models are more accurate than classification-based models in extracting highlights. This is likely due to the fact that regression allows us to consider fine-grained score assigned to each sentence in the article when identifying the highlights. In contrast, classification models require the assignment of labels at the sentence level, which results in a more coarse-grained assignment of scores to sentences.

4.4 Exploiting pivot words to model discourse facets in scientific papers

Results-oriented summaries of scientific papers play an important role in scientific communication. They allow readers to ascertain whether a paper is relevant to their own research by providing a summary of the paper's key contributions. However, different readers may be interested in different kinds of information in a paper. Researchers interested in the methodological detail of a paper will prefer a summary that emphasizes the section describing the research methodology, while researchers interested in the scientific contribution of a paper will favor a summary that emphasizes the section discussing the paper's results.

Providing a richer overview of a scientific paper requires the use of scientometric models to identify and summarize the different aspects of the document, such as the overview of the state of the art, the main findings and future directions. Results-oriented summaries focus on the most significant findings of the article, while facets-oriented summaries also highlight additional aspects of the paper, thus providing a more complete overview. We address the problem of generating facets-focused summaries of scientific documents by decomposing the task into two sub-tasks: discourse facet classification and discourse facet summarization [84].

Discourse facet classification The task of discourse facet classification has been originally proposed as part of the CL-SciSumm shared task [22]. Considering a reference paper R and a set of citing papers that cite it, the task is to label each citation c with the discourse facet it belongs to. In this case, each citation is provided with a snippet of the paper text corresponding to the citation context. The task requires to label each citation context with a discourse facet, depending both on the information conveyed by the context and the content of portion of the paper text the context refers to. The discourse facets that can be associated to a citation context c are: *Method*, *Results*, *Aim*, *Implication*, and *Hypothesis*.

We frame the task of discourse facet classification as a text classification problem. We define a set of features for each citation c and we use them to train a machine learning model to label c with a discourse facet.

Discourse facet summarization The task of discourse facet summarization consists in generating summaries for a scientific paper p that are oriented to the different discourse facets identified in the paper. The idea is to provide a set of summaries that focus on different aspects of the paper p. In particular, given a set of discourse facets F, the facet summarization task entails the extraction of a summary for each discourse facet in F.

We use supervised extractive summarization to automatically generate summaries for the different discourse facets of a reference paper. Specifically, our approach relies on regression models that learn to predict the importance of the sentences in the reference paper, with the aim of identifying the most salient sentences for each discourse facet. To train our models, we use the same set of features that we use to train the models for discourse facet classification.

4.4.1 Feature engineering

The machine learning models that we train for the tasks of discourse facet classification and summarization are based on a set of hand-crafted features. These features are aimed at capturing different aspects of the citation context that are relevant for the prediction of the discourse facet as well as the relevance of a sentence with respect to a particular facet.

Most of those features rely on the concept of pivot words, which are defined as the most discriminative words that can be used to identify a particular facet. The occurrence of pivot words in the citation context or in the cited snippet is a strong signal of the presence of a particular facet [47]. Leveraging training data, we can automatically learn pivot words for each of the discourse facets.

Defining as R the reference paper, C the citing paper, $c_{\rightarrow} \in C$ the context of the citation and $c_{\leftarrow} \in R$ the cited text snippet, we extract features that can be grouped into three categories: structural-based, similarity-based and relevance-based. Those features are extracted for each pair of citation context and cited text snippet $(c_{\rightarrow}, c_{\leftarrow})$.

Structural features Structural features aims at capturing patterns in the position or in the length of the citation context and cited text snippet. In particular, we extract the following features:

- The relative position of $c_{\leftarrow}$ in R.
- The relative position of $c_{\rightarrow}$ in C.
- The length of $c_{\leftarrow}$ expressed in terms of number of words.
- The presence of symbols (e.g., non-aphabetical characters) in $c_{\leftarrow}$.
- The section in which $c_{\leftarrow}$ appears. To ensure consistency in the section labeling, we utilize the IMRaD classification scheme [168].

All features extracted from the position of the citation context and cited text snippet are normalized to the length of the overall document.

Similarity features Similarity features estimate syntactic and semantic coherence of the text span to the facet class by leveraging pivot words. Pivot words are learned using the methodology proposed by Fu et al. [47]. The pivot words for each facet are extracted from the training data and are subsequently used to compute the following similarity features:

- The number of pivot words appearing in $c_{\leftarrow}$. For each facet class we define a separate feature to compute the frequency of occurrences of the corresponding pivot words.

- The number of pivot words appearing in $c_{\rightarrow}$. Similarly to the previous case, we define a separate feature for each facet to compute the frequency of occurrences of the corresponding pivot words in $c_{\rightarrow}$.
- The semantic distance between words occurring in $c_{\leftarrow}$ and the pivot words of each facet. We use the Word2Vec model [123], trained on a large collection of scientific papers [33], to compute word embeddings and we estimate the score using Word Mover Distance [82] between $c_{\leftarrow}$ and the pivot words.
- The semantic distance between words occurring in $c_{\rightarrow}$ and the pivot words of each facet. We follow the same strategy employed for the semantic distance of the cited text snippet.

Those features are designed to capture the lexical and semantic coherence of the cited text snippet and citation context with the discourse facet. The intuition is that, if the cited text and citation context are semantically and syntactically coherent with a facet class, then it is likely that their content belongs to that facet class.

Relevance feature This feature estimates the overall relevance of the cited text span within the reference paper. For each reference paper R we generate a graph $G_R = (V_R, E_R)$ whose nodes V_R are the sentences in R (including $c_{\leftarrow}$), and the edges E_R are computed using the cosine similarity between the embeddings of the sentences in R. Sentence embeddings are computed using SciBERT [8] and the edge weights are computed using the cosine similarity between the sentence embeddings. The overall relevance of $c_{\leftarrow}$ is estimated using its PageRank score [138] within G_R.

Using a set of features covering various aspects of citing context and referenced text snippet, we aim at generating a comprehensive representation that can be used both to predict the discourse facet and to estimate the sentence relevance for summary generation.

4.4.2 Discourse facet classification

Given a reference paper R and a set of citing papers C, the task is to label the pair consisting of the citation context $c_{\rightarrow} \in C$ and the cited span $c_{\leftarrow} \in R$ with a label l representing the discourse facet of the citation. We use the features described in

Section 4.4.1 to train a Gradient Boosting classifier to predict the discourse facet of the citation.

Dataset To train and evaluate the machine learning models we use the data collection released by CL-SciSumm organizers [22]. It contains 40 reference papers and the corresponding annotations of the discourse facet for each citation. It contains 40 reference papers and contains the annotation of the discourse facet for each citation. It also contains the citation context in the citing paper and the identifier of the cited snippet in the reference paper. We select 75% of the reference papers to generate training examples and 25% for testing. To reproduce our results and to allow for a fair comparison with future work, the train/test splits are made publicly available for research purposes[3].

Pivot words are crucial for the effectiveness of the proposed methodology. To extract the most relevant pivot words both in citing and cited snippets we leverage the open-source implementation of the mining algorithm proposed by Fu et al. [47][4]. It capitalizes on the idea that the occurrence of specific words is strongly correlated with the text attribute class. In our context, pivot words are words that are strongly correlated with a specific discourse facet.

Evaluation metrics We adopt the standard evaluation metrics for classification tasks, i.e., precision, recall, and F1-score. The precision describes the percentage of correctly classified citations out of all the considered citations, whereas the recall is the percentage of correctly classified citations out of all the citations that should have been classified as a specific label. The F1-score is the harmonic mean of precision and recall. Since the discourse facet classification is a multi-class classification task, we report,

- the macro average, i.e., the average of the metrics over all the classes;
- the micro average, i.e., the metric over all the classes computed considering true positives, false positives, and false negatives with respect to all classes.

[3]https://github.com/MorenoLaQuatra/Auto-Scientific-Annotation
[4]https://github.com/FranxYao/pivot_analysis

Table 4.4 Results of the system comparison for the discourse facet classification task. Best results are highlighted in boldface.

System	Metric type	Precision	Recall	F1-score
Poli2Sum	Micro Average	0.59	0.59	0.59
	Macro Average	0.24	0.22	0.23
	Weighted Average	0.77	0.59	0.67
CIST	Micro Average	0.67	0.67	0.67
	Macro Average	0.24	0.24	0.23
	Weighted Average	0.81	0.67	0.73
Pivot-Based Classification	Micro Average	**0.77**	**0.77**	**0.77**
	Macro Average	**0.36**	**0.55**	**0.41**
	Weighted Average	**0.88**	**0.77**	**0.81**

- the weighted average, i.e., the metric over all the classes computed by weighing the scores for each class according to their support (the number of instances for each label);

System comparison We compare the performance of our system, namely *Pivot-Based Classification*, with our previous system designed for the CL-SciSumm shared task (Poli2Sum) [83] and the best-performing system presented in the shared task (CIST) [94]. Similarly to Pivot-Based Classification, Poli2Sum relies on machine learning classifiers trained on a set of hand-crafted features but it does not consider pivot words for classifying the discourse facets of the citations. CIST, instead, relies rule-based classification models that generate a set of rules for categorizing the discourse facets of citation considering both the citing and the cited sentences.

Discourse facets have different distributions in the CL-SciSumm dataset. The *method* facet is the most frequent with 69.6% of the citations, followed by *result* (12.9%), *aim* (7.7%), implication (7.4%) and *hypothesis* (2.2%). Using machine learning-based classifiers to predict the *hypothesis* facet is practically unfeasible since there are too few training examples (i.e., 18 over 808 samples). To better compare the systems, we describe the performance for the four most frequent facets.

Table 4.4 reports the results obtained by the three systems on the test set. From the table, we observe that our system outperforms both Poli2Sum and CIST, achieving the best results in all the considered evaluation metrics. This indicates that pivot words are effective for the identification of the discourse facets of a citation and the

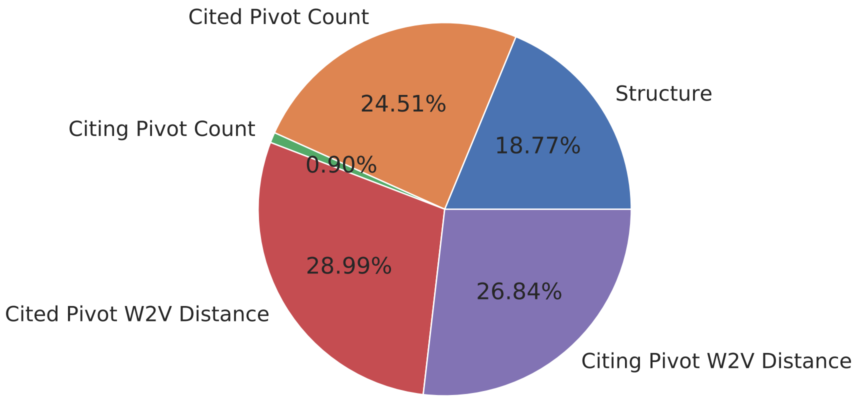


Fig. 4.6 Feature importance analysis in discourse facet classification.

set of features used by our system is able to capture a more comprehensive set of information to classify the discourse facets of a citation.

Feature relevance analysis In addition to standard evaluation metrics, we perform a feature relevance analysis to assess the relevance of the features used to classify discourse facets. Towards this end, we examine the feature importance scores provided by the trained classification model. To gauge the importance of each feature type we aggregate feature importance scores for all features of a specific type and for all facet classes, thus obtaining the probability that the feature type is relevant for the overall classification outcome.

Figure 4.6 shows the distribution of feature importance scores for features of different types. Overall, features related to pivot words occurrence (count) or semantic distance are the most relevant both considering the citing context and the cited snippet. This is consistent with the argument that the presence of pivot words is a reliable indicator of the purpose of the citation and that the cited snippet generally contains words that are semantically close to the pivot words. The number of occurrences of pivot words in the citing context, however, is not considered a relevant feature by the classification model. The reason may be that the simple count of pivot words in the citing context is weakly correlated with the target class because noise in the input data might affect it.

4.4.3 Facet-specific summarization

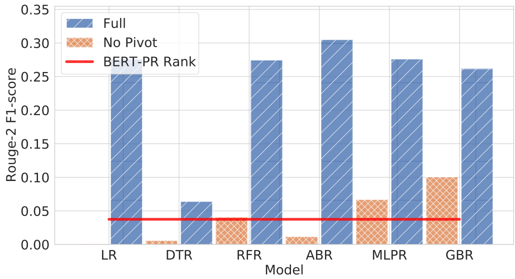
Generating facet-specific summaries for scientific publications requires both the identification of the sentences that pertain to a specific discourse facet and the ability to select the most salient ones to include in the summary. Given a reference paper R and a set of papers citing R, the goal is to extract different summaries, one for each discourse facet. Each summary consists of a set of sentences extracted from R.

We address this task by adopting a supervised extractive summarization approach that is trained to rank sentences in a document according to their importance for a specific discourse facet. Given a discourse facet $f \in F$, for each sentence in R we extract the features described in Section 4.4.1 and train a regression model to predict the importance of the sentence for the discourse facet f that we want to summarize.

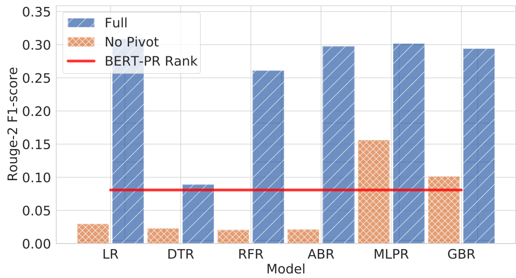
The label for the regression task is obtained by measuring the ROUGE-L [97] score between each sentence in R and the reference summary. During inference, the sentences are ranked according to the importance scores predicted by the regression model and they are iteratively included in the summary until the threshold on length of the summary is reached.

Data collection The dataset proposed for the CL-SciSumm shared task contains, for each reference paper, a *community summary* including the most important sentences for each discourse facet. We split each summary into four different summaries by leveraging sentence-level labels that identify the discourse facet for each sentence. This allows us to obtain a labeled dataset consisting of reference papers that we want to summarize and a set of reference summaries, one for each discourse facet. Similarly, for the evaluation of the system on the test set, we split the *community summary* of each reference paper into four different summaries and use them as references for the evaluation.

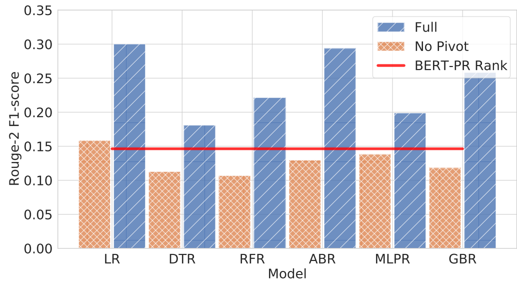
To evaluate the summaries generated by the system, we use the ROUGE-2 F1-score between the system's summary and the reference summary, separately per discourse facet. To assess the effectiveness of the proposed approach, we tested and compare a range of different regression models, including Linear Regression (LR), Decision Trees (DTR), Random Forest (RBR), AdaBoost (ABR), Multi-Layer Perceptron (MLPR), and Gradient Boosting (GBR). All the models are trained on the same set of features and labels, and they are evaluated on the same test set. To



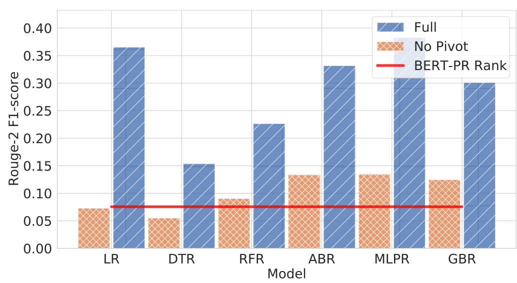
(a) Aim facet



(b) Implication facet



(c) Method facet



(d) Results facet

Fig. 4.7 Results comparison for the discourse facet summarization task.

further validate the effectiveness of integrating pivot words in the proposed regression models, we evaluate each of them with (Full) and without (No Pivot) the features extracted using pivot words.

Wa also compare the models with an unsupervised baseline (BERT-PR Rank) that extract the summary by computing the overall importance of each sentence according to the PageRank [138] algorithm. Specifically, each paper is modeled as a directed graph, in which the nodes represent its sentences and the edges' weights are computed according to the cosine similarity between the sentence vectors obtained using SciBERT [8]. The sentence relevance is computed by running PageRank on the graph, and the sentences are then ranked based on their PageRank score.

Figure 4.7 shows the ROUGE-2 F1-score for each model separately on each discourse facet. Comparing *Full* and *No Pivot* results, we can observe that the pivot words are effective in all the models and improve the ROUGE-2 F1-score significantly. Further, we observe that, in the majority of cases, the removal of the pivot words results in lower ROUGE-2 F1-score than the unsupervised baseline, supporting the value of these additional features.

Linear Regression, AdaBoost and Multi-Layer perceptron models show the best results among the supervised models. While MLPR outperforms the other models on the *implication* and *results* facets, the Linear Regression model shows the best results on the *method*. Overall, the majority of models trained with the full set of features are able to reach a ROUGE-2 F1-score of more than 0.25, whereas the unsupervised baseline (BERT-PR Rank) reaches ROUGE-2 F1-scores in the range of 0.03-0.15, depending on the discourse facet.

The promising results obtained both for the discourse facet classification and summarization tasks support the hypothesis that the proposed features are effective in modeling the relation between a citation context and its corresponding referenced text snippet. To foster further contributions in this direction, we make (i) the list of pivot words, (ii) the facet summaries, (iii) the automatic facet assignments predicted by our models and (iv) some qualitative examples are available to the community [5]. Specifically, we release the automatic facet assignments (i.e., the prediction of the discourse facet classification model) for all the papers of the ScisummNet dataset [197], a larger data collection that does not include the labels of the discourse facets.

4.5 Unsupervised slide generation from academic papers

Academic publications are usually presented at scientific conferences in the form of a talk, which is typically accompanied by slides. The slides are intended to be used during conference presentations, to supplement the live explanation of the presenter. Slide generation is traditionally a manual process that requires a significant amount of time and efforts. The authors of scientific papers must manually select and prepare slides that provide an overview of their research work.

This section discusses a technique for automatic slide extraction from academic papers, which does not require training data [17]. While general-purpose summaries for scientific papers are widely available [29, 30, 56], the specific case of slide generation from academic papers has received less attention in the literature. We formulate the slides generation problem as an extractive summarization task. Slides are expected to be shorter than the full paper and each slide should be focused on

[5]https://github.com/MorenoLaQuatra/Auto-Scientific-Annotation

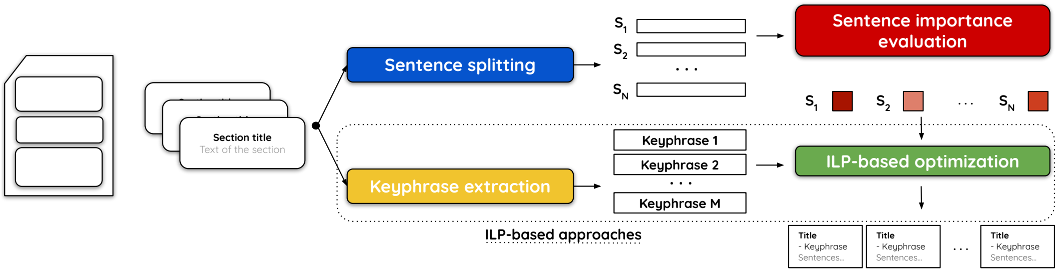


Fig. 4.8 Sketch of the slides extraction pipeline.

a particular aspect of the paper. Previous approaches addresses the problem using supervised learning methods [67, 165] that require a training dataset of academic papers and their corresponding slides. In contrast, we investigate the use of an unsupervised method that does not require annotated training data.

4.5.1 Proposed pipeline

Our unsupervised pipeline for slide generation is shown in Figure 4.8. The process consists of the following steps, which are performed using standard NLP tools and custom-defined heuristics:

1. *Pre-processing*: this step is required to standardize the input data format. The full text of each paper is divided into sections, and for each section, the text is broken up into sentences.

2. *Sentence importance assessment*: we tested several ranking methods to estimate sentence importance. They assign a score to each sentence indicating its overall relevance in the publication.

3. *Keyphrase extraction*: this is an optional step that extract noun phrases from paper's sentences. Slide content is usually organized in bullet points. Each bullet point typically contains a keyphrase that summarizes an important concept. In this step, we extract *noun phrases* (i.e., a set of words including a noun and modifying elements) that can be used to better estimate sentence relevance for the main paper's contributions.

4. *Sentence selection*: we propose two different unsupervised pipelines for selecting sentences.

- *ILP-based pipeline*: this approach exploits overall sentence relevance and jointly maximize (i) sentence diversity, (ii) sentence importance, and (iii) relevance of key phrases. The optimization step is performed using integer linear programming (ILP).
- *Summarization-only pipeline*: this method simply leverages unsupervised algorithms to rank and select the most relevant sentences.

5. *Slides generation*: the last stage in the pipeline aims at generating the presentation slides. By combining the information gathered at the previous steps, presentation slides are generated according to established guidelines [67].

4.5.2 Evaluation metrics

In order to evaluate the performance of the proposed approach, we use the ROUGE evaluation metric [97]. Similarly to previous studies [67, 165], we use the F1-measure of ROUGE-1 and ROUGE-2 scores to measure the overlap between the generated slides and the ground truth. However, the content of the slides needs further analysis in order to determine its adequacy for a presentation. We therefore proposed a new metric tailored toward the slide generation problem. Specifically, we consider the coverage of specific facets of the paper during slides evaluation.

The content of the ground-truth slides is first categorized into 4 classes following the IMRaD scientific paper structure [168] (e.g., Introduction, Method, Results and Discussion). Then, we use ROUGE scores to compare the generated slides with each category of the ground truth. Our goal is to provide an in-depth analysis of slides generation approaches that not only focuses on the overall overlap between generated slides and ground-truth slides, but also considers whether the slides cover all essential paper content. We denote the proposed metrics as *facet-specific* ROUGE scores because they can be used to evaluate the performance of slides generation approaches according to specific discourse facets of the paper.

4.5.3 Experimental results

To assess the performance of the proposed unsupervised pipelines we evaluate three pipeline configurations:

1. *Supervised ILP-based pipeline*: a set of regression models are trained using supervised learning to estimate sentence relevance. Specifically, we tested Support Vector Machines (SVR), MultiLayer Perceptron (MLP), Gradient Boosting (GB), Decision Tree (DT), and Random Forest (RF). Each model is trained on the features proposed by PPSGen model [67]. The estimated relevance scores are then used to obtain the final paper's slides using ILP that jointly optimize the objectives discussed in the previous section.

2. *Unsupervised ILP-based pipeline*: we use several unsupervised summarization approaches that are able to provide a relevance score for each sentence in the paper. Similarly to supervised approaches, those scores are used as input for the ILP optimization stage. Specifically, we use LSA-based methods (e.g., ELSA [15] and LSARank [171]), graph-modeling approaches (e.g., TextRank [122] and LexRank [41]) and models leveraging semantic sentence embeddings [153] (e.g., Centroid-BERT [89]).

3. *Unsupervised summarization-only pipeline*: the performance of standard unsupervised models has been evaluated using the same approaches used in the previous version, but without integrating ILP optimization objectives.

The slides generation task depends on the availability of a dataset of papers equipped with their corresponding slide decks. Our evaluation was based on a real data collection of 195 academic papers and their corresponding presentation slides[6]. Slides in the data collection have been manually annotated by the authors of the corresponding papers. We use the full data collection as test set to evaluate the proposed unsupervised pipelines, while, for the supervised methods we train the regression models using leave-one-out cross-validation (LOOCV).

Table 4.5 summarizes the results of the evaluation in terms of standard and facet-specific ROUGE scores. As expected, supervised methods outperform unsupervised approaches in standard ROUGE metrics, since the latter are based on a general-purpose sentence retrieval models, which cannot be trained on the specific task of slide generation. On the other hand, ILP optimization is able to significantly improve the performance of unsupervised models, without requiring a supervised training phase.

[6]The data collection is available upon request at https://github.com/hairav/SlideSpawn (latest access: June 2022)

Table 4.5 ROUGE scores obtained using the three configurations of the proposed pipeline. The results are reported for the overall evaluation as well as all the considered facets. The highest scores, for each column, are highlighted in **bold** and statistically significant improvements ($p < 0.05$) are marked with *.

	Overall		Introduction		Method		Results		Discussion	
Approach	R1-F	R2-F	R1-F	R2-F	R1-F	R2-F	R1-F	R2-F	R1-F	R2-F
Supervised ILP-based pipeline										
MLP	0.256*	0.106	0.035*	0.000*	0.035*	0.001	**0.044**	**0.001**	0.053	**0.003**
SVR	0.257*	0.105	0.037*	0.003	0.038*	**0.002**	**0.044**	**0.001**	0.048*	0.002
GB	0.258	0.104	0.042	0.003	0.040	0.001	**0.044**	**0.001**	0.053	**0.003**
DT	0.248*	0.096*	0.037*	0.003	**0.045**	0.001	**0.044**	**0.001**	0.051*	0.002
RF	**0.263**	**0.109**	0.032*	0.003	0.038*	**0.002**	0.043	0.000	0.046*	0.002
Unsupervised ILP-based pipeline										
ELSA	0.241*	0.089*	**0.045**	0.003	0.041	0.001	0.041	0.000	**0.06**	**0.003**
Centroid	0.233*	0.078*	0.026*	0.000*	0.032*	0.000*	**0.044**	0.000	0.052*	**0.003**
LexRank	0.233*	0.078*	0.026*	0.000*	0.032*	0.000*	**0.044**	0.000	0.052*	**0.003**
TextRank	0.232*	0.078*	0.026*	0.000*	0.032*	0.000*	**0.044**	0.000	0.052*	**0.003**
LSARank	0.223*	0.065*	0.042	0.003	0.036*	0.001	0.024*	0.000*	0.059	**0.003**
Unsupervised summarization-only pipeline										
ELSA	0.218*	0.060*	0.033*	0.001	0.026*	0.000*	0.032*	0.000	0.033*	0.001*
Centroid	0.230*	0.070*	0.041	**0.004**	0.020*	0.000*	0.026*	0.000*	0.040*	0.001
LexRank	0.218*	0.058*	**0.045**	0.001*	0.024*	0.000*	**0.044**	**0.001**	0.054	0.000*
TextRank	0.214*	0.060*	0.029*	0.001*	0.025*	0.000*	**0.044**	**0.001**	0.029*	0.000*
LSARank	0.220*	0.064*	0.026*	0.001*	0.026*	0.000*	0.036*	0.000	0.032*	0.001*

Considering facet-specific ROUGE metrics, we observe that the unsupervised ILP-based pipeline perform on-par or outperforms the supervised ones in the *Introduction* and *Discussion* facets, while the latter are able to provide better scores in the *Methods* and *Results* ones. We believe that this could be probably due to the fact that the slide generation task is easier for the *Introduction* and *Discussion* since these sections usually contain general discussions about the paper's topic and do not include specific details or mathematical formulae. In contrast, the *Methods* and *Results* sections contain specific details about the paper's topic, which are more difficult to summarize and require the use of a supervised model. This observation is further confirmed by the results of the unsupervised summarization-only pipeline, which, compared with the unsupervised ILP-based pipeline, is able to provide similar or better results for the *Introduction* facet.

4.6 Future research directions

Researchers have made great progress in mining scientific publications to extract useful knowledge and information. The automated analysis of scientific citations and semantic understanding of publications' content represents an important and valuable area of research that can have a significant impact on the way we perform research. Both NLP and Scientometrics are fields whose interconnected nature has generated considerable interest in recent years. Scientometrics, in particular, has a long history of using data from research publications to understand and identify trends. In recent years, however, the availability of electronic publications and advances in NLP have led to new opportunities for integrating computational techniques into this discipline. The application of NLP can be used to complement and enhance traditional bibliometrics approaches by providing more accurate data to support analysis.

The citation context analysis, in particular, represents an important area of future research. The context of citations provides valuable information that can be analyzed to better understand the purpose of citations and how papers are related. Machine learning has proven to be successful in this domain. The semantic representations provided by modern language models allow to build accurate models that can effectively analyze and predict the purpose of citation.

This research line highlighted the importance of the interaction between Scientometrics and NLP. It can have many benefits that can help the automatic analysis of scientometric data. Even though NLP has been used successfully in many different scientometrics tasks, there are still many opportunities for improvement. Research papers have standardized structures that provide valuable information that can be used as an inductive bias in training large-scale deep learning models. Those models are usually trained leveraging self-supervised learning objectives that allow to learn useful representations that are transferable to other tasks. The use of *constrastive learning* [155] and self-supervised learning objectives for the automatic analysis of scientometric data is a promising direction for future research. Those techniques usually leverages the structure of texts and documents, thus they can be used to design specific self-supervised objectives to effectively train large language models without the need for manual annotations.

The task of automatic summarization of scientific documents is a challenging task that has many potential applications. Sequence-to-sequence transformer models [151, 92] are the current state-of-the-art in the field of abstractive text summarization. Despite several attempts to adapt those models to the analysis of long sequences [194], there are still many challenges that need to be addressed. One of the main limitations is the computational resources required to train those models. Therefore, future research should focus on novel architectures specifically designed for summarizing long scientific documents, as well as ways to reduce the computational cost of training these models.

Finally, the automated analysis of a scientific publication strongly relies on specific scientific concepts discussed in a paper. Recognizing domain-specific concepts is a challenging task, as the language used in scientific papers can be technical and often includes domain-specific vocabulary. The ability to better understand the language used in scientific documents can be instrumental in identifying emerging trends and potential research areas in a given field [64]. Identifying scientific concepts discussed in a paper can also be used to improve the generation of semantically-rich summaries of research papers. A key focus of future research should be on developing novel models that can effectively detect domain-specific scientific concepts discussed in a paper.

Chapter 5

Time matters in Text Summarization

NLP focuses on the analysis of user-generated content. It is primarily used to extract information from unstructured text and to support several end-user tasks (e.g., question answering, text summarization, or conversational assistant). Understanding the time dimension is crucial in this context, as it is a fundamental element of human communication. For instance, in politics [63], in business [164] and in social media [19] the temporal dimension is often crucial to extract useful information from user-generated content. The aim of this chapter is to provide an overview of methods and challenges related to temporal analysis in text summarization by grounding the discussion in a real-world scenario. Section 5.1 review the main studies in the field, exploring the effect of the time dimension in text summarization. Sections 5.2 and 5.3 outline our contributions in single and multi-lingual timeline summarization, respectively. Finally, Section 5.4 explores the main challenges in the field and potential future developments. A graphical overview of the contributions presented in this chapter is shown in Figure 5.1.

5.1 Temporal information in text summarization

Temporal information is often present in user-generated content. Political speeches, social media posts and news articles often contain references to time-related content to emphasize a specific event and provide contextual information. The temporal dimension is often crucial in the process of gathering, understanding and presenting information to a human audience. Text summarization aims at extracting the most

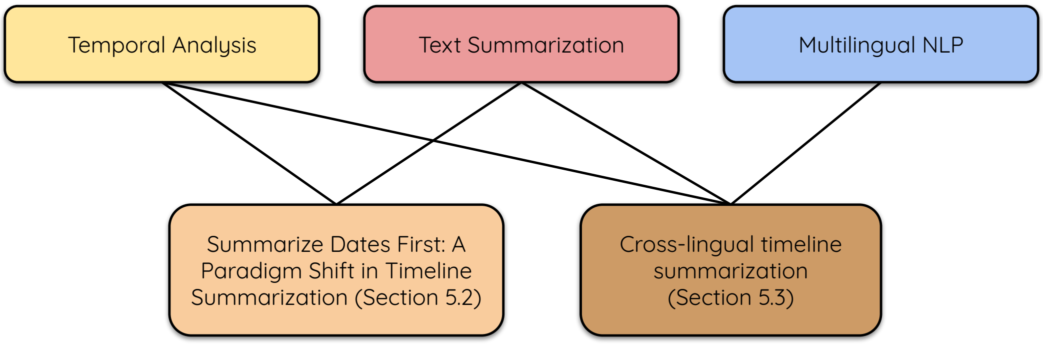


Fig. 5.1 Graphical overview of the topics and contributions covered in Chapter 5

relevant information from source texts and presenting it in a concise way. The concept of time is often crucial in this process as it may provide the user a concise understanding of the most relevant events, their temporal ordering and the temporal relationships between them.

Temporal and timeline summarization are two tasks designed to provide relevant information from a collection of documents by focusing on the chronological order of events. They are both considered sub-tasks of multi-document summarization since they both require the identification of the most relevant information from a collection of documents pertaining to a specific event.

5.1.1 Temporal summarization

Temporal summarization aims at finding and extracting information about specific events. It is commonly used for monitoring purposes to provide timely information about the most relevant happenings (e.g., accidents or natural disasters). Compared to the standard summarization task, it entails the analysis of document streams and the temporal ordering of events. Each document in the stream contains time-related information (e.g., a publication time stamp) and refers to a specific event. The main task is usually broken-up in two different subtasks:

1. *Filtering*: the event is represented by a text query. The objective is to identify all documents in the document stream that contain relevant information (i.e., those that provide information about the event).

2. *Summarization*: the objective is to *emit* novel information once it becomes available from an event stream (i.e., those documents that have been retained in the previous step), disregarding the information that has already been emitted at previous timestamps.

The task was originally proposed at the TREC conference [1]. The main goal is to detect and track events as they unfold (i.e., in real-time or near-real-time). For example, the query *Buenos Aires rail disaster* [1] can be used to follow the progress of the disaster in the document stream, consisting in a set of news articles published over the course of several days. The *filtering* step is focused on identifying documents that contain relevant information about the disaster (e.g., articles reporting information about the event), while the *summarization* step is focused on generating live updates about the disaster as more and more information becomes available (e.g., new information about the death toll, or the causes of the disaster). The task participants were asked to address one or both of the two main subtasks: filtering and summarization. The temporal information, in this case, is typically represented as a timestamp and it is used to organize the information in a stream. The majority of the approaches apply traditional information retrieval techniques to identify documents containing event-related information and incrementally estimate their novelty using both the publication timestamp and previously emitted summaries [118, 207]. Throughout the years, the original task has been extended to include microblogging data [101] or updated event streams [117].

5.1.2 Timeline summarization

News articles and microblogging data provide valuable temporal information both considering their publication dates and their contents. Given an interest topic, the timeline summarization (TLS) task aims at generating a timeline of events that happened around the topic, where each event is described by a short summary. Unlike temporal summarization, this task does not focus on real-time updates for specific events, rather it provides an overview of the events that happened around a topic, thus allowing the users to have a global perspective of the events. For example, while temporal summarization may be interested in real-time updates about a specific episode (e.g., *Buenos Aires rail disaster*), timeline summarization could be interested in generating a timeline of a topic that usually spans over longer time spans (e.g., *COVID-19 pandemic*). The process involves the selection of salient dates and the creation of summaries explaining the events that occurred on these dates. Therefore, the timeline contains important dates, as well as a description of the events connected to them, thus providing a holistic viewpoint.

The interest on timeline summarization has been growing over the last few years as it can be useful for several applications. It can be used, for instance, for automatic content curation, providing an overview of events that happened around a specific topic, or to retrieve the most influential events over time by considering the most significant ones from a temporal perspective [188]. The TLS task aims to reach two different goals:

- *Select salient dates*: find the most important dates for a specific topic, which can be used to illustrate the evolution of a phenomenon over time. It should consider all relevant events as well as their temporal ordering.
- *Date summarization*: for each selected date, summarize the most important events that occurred around that date. It leverages the text content of the source documents to provide a summary of the most relevant events.

Clearly, the extraction of temporal information from text is essential to tackle the task of timeline summarization. It can be used both for assessing the relevance of dates as well as linking events to specific dates (e.g., sentences in a news articles could refer to events that occurred in the past or will take place in the future). The body of research in timelime summarization has focused on the design of methodologies to address one or both of the tasks above.

The selection of salient dates has been investigated as an independent task. In this case, the goal is to determine the most important dates for a given topic via information retrieval (IR) approaches. The re-ranking methodology proposed by Kessler et al. [76] identifies date references in text content and uses them to train a machine learning model that can estimate date relevance according to a specific set of features. The system uses annotations provided by journalists and domain experts to train a supervised classifier. However, large-scale annotated data is not always available, and therefore relying on supervised learning is not feasible in many cases. The same problem has been addressed by unsupervised approaches leveraging graph-modeling techniques [181]. Dates are modeled as nodes in a graph and the links between nodes are computed according to article publication date and in-text references.

Using text summarization techniques, extractive and abstractive approaches are used to generate dates summaries. The task is usually modeled as a multi-document summarization task, where the intent is to generate a summary that contains information extracted from different documents. The methodologies to address the summarization task in TLS use topic modeling for generating an overview of the event timeline [175] or combines user-specified queries with topic identified in news articles [26].

To generate a timeline, which requires the identification and summarization of salient dates, the single task of date selection or summarization is not sufficient. Therefore, existing approaches tackle both tasks either through a joint optimization formulation [112, 109] or by proposing a sequential pipeline leveraging unsupervised [133, 199, 96] or supervised learning [11, 51] for date selection. For each topic, the model outputs include a list of dates relevant to that topic along with brief summaries that highlight the most important events.

5.2 Summarize Dates First: A Paradigm Shift in Timeline Summarization

This section introduces our contribution to the field of timeline summarization. Our approach is based on the idea that we should *summarize dates first (SDF)*, meaning that for each candidate date, we extract a short summary highlighting the main events that occurred during that timeframe [86]. Once the summaries for all dates have been extracted, we select the most relevant dates as the ones that should be highlighted in the timeline. Comparing the proposed pipeline to the existing approaches, it has three main differences. First, compared with joint optimization approaches [112], it address the timeline summarization task by decoupling the summary generation from the date selection tasks. Second, it reverses the order on how dates are summarized and selected, meaning that it generates summaries for all dates before selecting the ones that are most relevant to the timeline. This allow us to leverage date summaries during the selection process. Third, compared with supervised approaches [51], the

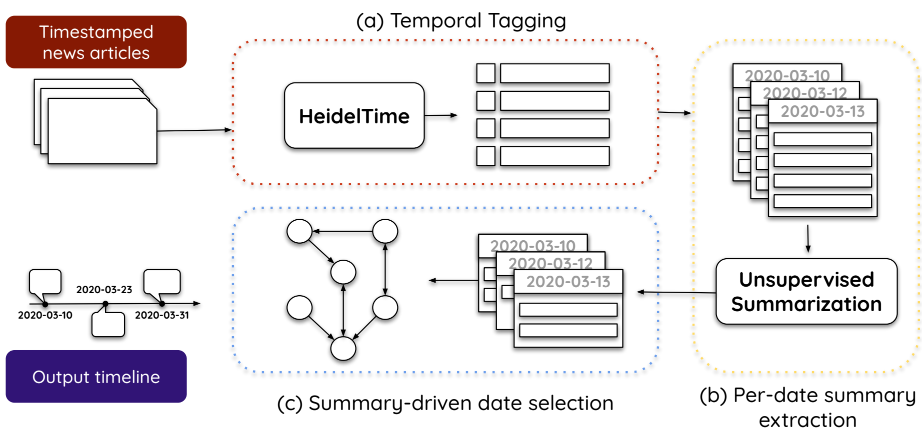


Fig. 5.2 Sketch of the proposed model for the Timeline Summarization task.

proposed pipeline is fully-unsupervised, thus it does not require the annotation of a gold-standard timeline to train the model.

5.2.1 Timeline summarization pipeline

Figure 5.2 shows a sketch of the proposed model. The pipeline takes in a set of documents with their publication timestamps and generates a timeline consisting of a short summary for each selected date. It is composed of a 3-step process:

(a) *Temporal Tagging*: the text of each publication is processed with HeidelTime [173], a rule-based system that detect and normalize temporal expressions in the text content (e.g., it converts "10th of March 2020" to "2020-03-10"). Each temporal expression is converted to a single timestamp and associated to the span of text in which it occurs. At this stage, we associate each sentence (i) to the referenced date, if a date is mentioned, (ii) to the publication date, otherwise.

(b) *Per-date summary extraction*: this step leverages unsupervised summarizer to extract the most relevant sentences for each date. In detail, for each date, we consider the set of sentences that contain a timestamp referring to that date and we use the summarizer to extract the most relevant sentences. It is worth noting that, the summarization step is applied to all dates, i.e., date selection is not yet performed at this stage. The output of this step consists of a set of dates with a corresponding summary provided as a list of relevant sentences.

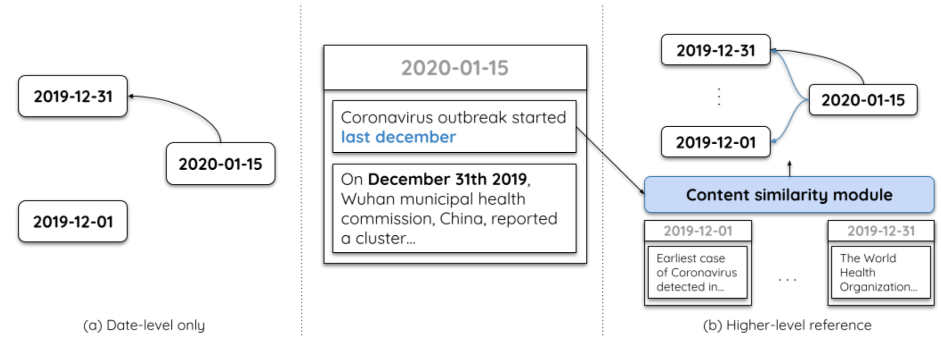


Fig. 5.3 Comparison of standard date-level references parsing and our proposed high-level graph enrichment.

(c) *Summary-driven date selection*: this step takes in the set of date-summary pairs, and it performs a selection of the most relevant dates. The selection step leverages graph modeling techniques. Each date is represented as a node in a directed graph and the edge between two dates is weighted according to,

- the temporal references, detected at step (a), from one date to another,
- the similarity of their corresponding summaries extracted at step (b).

To estimate the relevance of a date we experiment with different graph-centrality algorithms (e.g., node degree or PageRank [138]).

Through the decoupling of summarization and date selection steps, the system can be easily customized, as different summarization techniques can be applied and the selection process can be tailored to support different use cases. The inversion of the order of the selection and summarization steps in the pipeline allows the selection process to access the summary of each date, thus it can be used during the date selection to identify the most relevant dates.

High-level temporal references The TLS task aims at selecting relevant dates at day-level, thus temporal references for estimation of single date relevance are usually restricted to this level of granularity. However, by accessing the date summary, it is possible to exploit high-level temporal references (e.g. references to months or years). Given a sentences s_i published on a date d_i and containing a temporal expression referencing a certain month m, the system can extract the set of all dates $d_j \in m$ and compute the similarity of the summary of each date d_j with the sentence s_i. The similarity score is used to enrich the weights of the edges connecting d_i and d_j in the graph. An example comparing the standard parsing of temporal references and our proposed graph enrichment is reported in Figure 5.3. It is possible to note that by exploiting the high-level temporal references (HLTR) in the sentence, the system could update graph weights by exploiting a wider range of temporal references.

5.2.2 Benchmark TLS datasets

The proposed pipeline does not require supervision nor training data for the generation of the timeline. Therefore, we evaluated our pipeline on four benchmark datasets for timeline summarization:

- *Timeline 17* [182]: the dataset contains 19 manually-annotated timelines covering 9 different topics. Each topic is covered, on average, by 36.42 articles and the average timeline length is 242.47 days. It covers a variety of topics including: Egyptian protest, H1N1 outbreak, Haiti earthquake and other global events.

- *Crisis* [180]: this data collection covers 4 long-span armed conflicts (i.e., Egyptian revolution, Libya war, Syria war and Yemen crisis). Each topic is covered by $4,560.75$ articles on average and the average timeline length is 387.86 days.

- *Entities* [51]: the dataset covers 47 entities from the news domain, including people and organizations. Each entity is covered by $1,086.49$ articles on average and the mean coverage period is 19.02 years.

- *Covid-TLS* [86]: we propose a new dataset covering the COVID-19 outbreak. The dataset contains a single timeline with $26,376$ articles and a coverage period of 266 days.

For all the benchmark datasets, more than 50% of the references detected by HeidelTime [173] are not date-level references, thus the proposed graph enrichment could provide a significant contribution to the system performance.

5.2.3 Model configurations

The model proposed to address the TLS task leverages graph modelling to estimate the importance of dates and unsupervised summarization models to extract single-date summaries. The date selection process is guided by a directed graph $G(N,E)$ generated for each topic. Graph nodes $n_i, n_j \in N$ represent dates and the edges from n_i to n_j are weighted according to the temporal references and the similarity between their respective summaries.

To enrich the graph using high-level temporal references (HLTR), we use the syntactic similarity between the sentence containing the high-level temporal expression and the summary of the target date. Specifically, we use ROUGE-2 precision to compute the similarity between two text snippets. The choice of this similarity function is motivated by the need to compare text of different lengths, and by the fact that we aim at identifying overlapping of specific event information (e.g., proper nouns or locations). However, this approach can be seamlessly extended to other similarity metrics (e.g., semantic similarity using contextualized text embedding). In our experiments, we only use high-level references for months to reduce the inference costs and avoid data sparsity.

Once the graph $G(N,E)$ is generated using explicit temporal references and graph-enrichment techniques, the salience of each date can be estimated using different graph centrality algorithms.

- *PageRank[138]*: it estimates the importance of a node according to the number of links pointing to it and the importance of the nodes that point to it. This algorithm has been originally proposed to estimate the importance of a webpage in the World Wide Web, but it was later adapted to estimate the importance of any node of a graph. In the same way, we use it to estimate the salience of a date node in a directed graph generated from a date selection task.

- *HITS[79]*: it is a centrality estimation method that computes two different relevance scores, one for *hubs* and one for *authorities*. Hubs are nodes that point to relevant nodes, while authorities are pointed by relevant nodes. In our settings, we consider the *authority score* because we aim at finding those dates that are *referenced by* many other relevant dates.

- *Degree centrality*: the degree of a node is defined as the sum of the weights of incoming and outgoing connections. We consider both standard *degree* score and the *in-degree* of a node that only looks at incoming connections, on the premise that dates referenced by many others are likely to be relevant.

In our experiments, we used the centrality measures outlined above to determine the salience of dates and report their performance for each benchmark dataset.

The generation of summaries for each date is a crucial step in the proposed pipeline. It serves both as a proxy to estimate high-level temporal references and to generate the summaries of dates that are included in the final timeline. The lack of large-scale annotated datasets hampers the development of supervised summarization models, thus we opt for unsupervised extractive approaches. Our study compares several methods that have produced effective results in the text summarization task.

- *Graph modeling*: we test TextRank [122], LexRank [41], CoreRank [178] and TextRank-BM25 [5]. The first two are classic graph-based methods while CoreRank [178] is an approach that combines the submodular optimization and graph modelling for extractive summarization. TextRank-BM25 [5] is a variant of the TextRank algorithm that uses the BM25 score [156] to compute the similarity between two sentences.

- *LSA-based approaches*: we use ELSA [15], a recently-proposed unsupervised method that combines latent semantic analysis and frequent itemsets mining to extract key sentences.

- *Embedding-based models*: we evaluate modified versions of SubModular [99], Centroid-Rank [147], Centroid-Opt [50], EmbeddingRank [138] that use contextualized sentence embeddings [153] to rank sentences.

5.2.4 Experimental evaluation

Timeline summarization is a challenging task that combines both the selection of relevant dates and the generation of a short summary for each of them. Standard metrics for evaluating summarization models, such as ROUGE [97], are not adequate for this task, since they do not consider the selection of the relevant dates. Therefore, ad-hoc evaluation criteria have been proposed in the literature in order to evaluate model performance [111]. Those metrics are variants of standard ROUGE and aim at simultaneously assessing the model's ability to identify salient dates and summarize them.

- *ROUGE-concat*: it merges date summaries both for the system and reference timelines and evaluates them using standard ROUGE measures. It replicates the standard ROUGE evaluation by discarding the date information and comparing summaries as plain text.
- *ROUGE-agreement*: it evaluates the quality of the proposed timeline by separately comparing summaries for each date. It assumes that both the system and the reference timelines should agree on the set of salient dates. It incorporates the temporal information by requiring an exact match for dates in reference and system timelines to be considered for the evaluation.
- *ROUGE-alignment*: it is based on the idea that dates in the system and reference timelines could not match exactly but still refer to the same event. To address this issue, this metric aligns dates first, and then compute the ROUGE scores by evaluating the system summary against the reference summary corresponding to the aligned date.

The metrics above quantify different aspects of the model performance, hence, we report results for all three of them.

Model comparison We evaluated the proposed methodology through an extensive set of experiments, including date selection and summarization results. We compare the proposed model against (i) an unsupervised baseline proposed by *Chieu & Lee* [26] that generates daily summaries leveraging custom temporal heuristics and information retrieval techniques, (ii) a state-of-the-art unsupervised model that defines ad-hoc constraints to jointly optimize date selection and summarization objectives [112] and, (iii) the DateWise approach that uses supervised regression algorithms to determine date relevance and unsupervised summarization to extract summaries from selected dates [51]. Optimal parameters for each method were determined separately, according to the dataset, using grid search.

Table 5.1 Performance comparison for the date summarization task. Best-performing results for unsupervised pipelines are highlighted in bold and statistically significant performance worsening are starred.

Model	Method	concat		agreement		alignment	
		R1-F1	R2-F1	R1-F1	R2-F1	R1-F1	R2-F1
Timeline 17							
Chieu & Lee [26]	U	0.275*	0.065*	0.028*	0.008*	0.057*	0.014*
TLS+reweighting [112]	U	0.383	0.092	0.094	0.025*	0.109	0.028
(our) Degree, TextRank-BM25	U	**0.401**	**0.101**	**0.106**	**0.033**	**0.120**	**0.035**
DateWise [51]	S-DS	0.385	0.097	0.107	0.032	0.120	0.035
Crisis							
Chieu & Lee [26]	U	**0.368**	0.066	0.028*	0.005*	0.051*	0.009*
ASMDS+DateRef [112]	U	0.333*	0.07	0.051	0.011	0.073	0.016
(our) In-degree-HLTR, TextRank-BM25	U	0.360	**0.073**	**0.064**	**0.014**	**0.086**	**0.018**
DateWise [51]	S-DS	0.347	0.075	0.071	0.023	0.089	0.026
Entities							
Chieu & Lee [26]	U	**0.275**	**0.053**	0.025*	0.011	0.038*	0.012
TLS+reweighting+DateRef [112]	U	**0.275**	**0.053**	0.039	**0.013**	**0.051**	**0.015**
(our) In-degree-HLTR, TextRank-BM25	U	**0.275**	0.052	**0.041**	0.011	**0.051**	0.014
DateWise [51]	S-DS	0.271	0.051	0.045	0.014	0.057	0.017
Covid-TLS							
Chieu & Lee [26]	U	0.203	0.021	0.008	0.001	0.017	0.001
ASMDS+TempDiv+DateRef [112]	U	0.249	0.036	0.028	0.001	0.03	0.001
(our) Pagerank-HLTR, TextRank-BM25	U	**0.439**	**0.076**	**0.062**	**0.011**	**0.072**	**0.012**
DateWise [51]	S-DS	0.318	0.038	0.036	0.005	0.040	0.006

Date summarization results Table 5.1 reports the comparison of different methods for the date summarization task. It reports the performance in terms of *concat*, *agreement* and *alignment* metrics using ROUGE-1 and ROUGE-2 F1-score. The proposed method performed best, on average, compared to the other unsupervised approaches. On Timeline 17 our approach outperforms or, at least, is comparable to the state-of-the-art supervised methods. Since there is only one timeline for Covid-TLS, we cannot train the DateWise model on it. We instead train it on the Entities dataset and test it on Covid-TLS. The results obtained by our method on this dataset are significantly better than all the others for all the evaluation metrics.

Date selection performance We evaluate the date selection task by leveraging information retrieval formulation of precision (P), recall (R) and f1-score ($F1$). Given the set of dates selected by the system ST_d and the corresponding set of ground-truth dates GT_d the metrics are defined as follows:

Table 5.2 F1-score performance comparison for the date selection task. Best-performing results for unsupervised pipelines are highlighted in bold and statistically significant performance worsening are starred.

Model		Type	Timeline 17	Crisis	Entities	CovidTLS
Chieu & Lee [26]		U	0.230*	0.166*	0.09*	0.176
Martschat & Markert [112]	ASMDS	U	0.531	0.278	0.163	0.685
	TLS-constraint	U	0.527	0.266	0.180	0.679
Proposed method	In-degree	U	0.549	**0.302**	**0.197**	**0.689**
	HITS	U	**0.553**	0.206	0.095	0.679
	Pagerank	U	0.537	0.175	0.161	0.623
	Degree	U	0.532	0.275	0.117	0.679
DateWise [51]		S	0.544	0.295	0.205	0.679

$$P = \frac{|ST_d \cap GT_d|}{|ST_d|} \tag{5.1}$$

$$R = \frac{|ST_d \cap GT_d|}{|GT_d|} \tag{5.2}$$

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{5.3}$$

In Table 5.2 we compare the date selection performance of the proposed model and the competitors using the f1-score; it is obtained by taking the harmonic mean of precision and recall, thus giving a better representation of the overall performance of the model. Our model achieves the best f1-score among all unsupervised approaches and outperforms supervised models in three out of four datasets. Among the proposed graph-based algorithms, the in-degree centrality achieves the best performance overall (i.e., it is the best performing method on the majority of the datasets).

Unsupervised summarization The proposed pipeline relies on unsupervised summarization models to extract date summaries. By fixing the date selection configuration, we investigate the summary quality extracted by the different unsupervised summarization models. Table 5.3 shows the average scores of the summaries extracted by each unsupervised summarization model. We evaluate the timeline extracted by each method using the same set of metrics proposed for the standard TLS task. The *TextRank-BM25* [5] method shows the best performance overall across

Table 5.3 Performance comparison of different unsupervised models. Best-performing results are highlighted in bold and statistically significant performance worsening are starred.

Summarizer	concat F1		agreement F1		alignment-m21 F1	
	R1	R2	R1	R2	R1	R2
Timeline 17						
TextRank	0.363*	0.084*	0.086*	0.023*	0.097*	0.025*
LexRank	0.370*	0.084*	0.088*	0.025*	0.100*	0.027*
CoreRank	0.371*	0.091*	0.092*	0.024*	0.105*	0.026*
TextRank-BM25	**0.401**	**0.101**	**0.106**	**0.033**	**0.120**	**0.035**
ELSA	0.389*	0.097	0.100	0.029	0.114	0.032
SubModular	0.367*	0.082*	0.086*	0.024	0.098*	0.025
Centroid-Rank	0.365*	0.082*	0.084*	0.023	0.096*	0.025
Centroid-Opt	0.372*	0.082*	0.084*	0.021*	0.097*	0.023*
EmbeddingRank	0.365*	0.084*	0.087*	0.022	0.098*	0.024*
Crisis						
TextRank	0.311*	0.058*	0.043*	0.009	0.062*	0.012*
LexRank	0.312*	0.056*	0.042*	0.009*	0.059*	0.012*
CoreRank	0.356	**0.075**	0.060	0.014	0.080	0.017
TextRank-BM25	**0.360**	0.073	**0.064**	0.014	**0.086**	**0.018**
ELSA	0.338*	0.064*	0.061	**0.015**	0.081	**0.018**
SubModular	0.337	0.057*	0.050*	0.009	0.068*	0.012*
Centroid-Rank	0.335*	0.056*	0.047*	0.008*	0.065*	0.011*
Centroid-Opt	0.337*	0.057*	0.049*	0.009*	0.068*	0.012*
EmbeddingRank	0.337	0.056*	0.048*	0.008*	0.066*	0.011*
Entities						
TextRank	0.238*	0.041*	0.030*	0.007*	0.040*	0.010*
LexRank	0.245*	0.043*	0.032*	0.008*	0.041*	0.010*
CoreRank	0.258*	0.049	0.038	**0.012**	0.048	**0.014**
TextRank-BM25	**0.275**	**0.052**	**0.041**	0.011	**0.051**	**0.014**
ELSA	0.258*	0.044*	0.036*	0.009	0.046*	0.011*
SubModular	0.249*	0.040*	0.031*	0.007*	0.040*	0.009*
Centroid-Rank	0.251*	0.042*	0.032*	0.008*	0.041*	0.009*
Centroid-Opt	0.251*	0.041*	0.032*	0.007*	0.041*	0.009*
EmbeddingRank	0.250*	0.041*	0.032*	0.007*	0.041*	0.009*
Sars-Cov-2						
TextRank	0.452	0.061	0.045	0.004	0.054	0.004
LexRank	0.460	0.066	0.052	0.005	0.061	0.006
CoreRank	0.381	0.053	0.044	0.003	0.051	0.004
TextRank-BM25	**0.438**	**0.077**	**0.063**	**0.011**	**0.072**	**0.012**
ELSA	0.427	0.065	0.048	0.005	0.056	0.005
SubModular	0.423	0.055	0.051	0.006	0.059	0.006
Centroid-Rank	0.417	0.053	0.050	0.005	0.058	0.006
Centroid-Opt	0.435	0.057	0.049	0.005	0.057	0.006
EmbeddingRank	0.426	0.056	0.049	0.006	0.057	0.007

all data collections. Except for traditional baselines (i.e., LexRank and TextRank) embedding-based models underperform all other methods. Those approaches struggle to cope with named entities, which are essential for extracting event-specific summaries that capture the main actors and events. Syntactically-grounded methods, leveraging word co-occurrences, are more effective since they include features that enable them to better model key terms and entities identifying the events.

5.3 Cross-lingual timeline summarization

The field of timeline summarization has been heavily dominated by English-only approaches in the past years. In many real-world scenarios, however, we need to analyze data written in other languages. In this context, we explore the problem of cross-lingual timeline summarization (CL-TLS) task. Given a set of news articles written in multiple languages and covering a specific topic, the goal is to automatically detect relevant dates and generate summaries for each of them in a target language. The proposed CL-TLS task can be seen as a generalization of the standard TLS task, where the input documents are multilingual and the output summaries are written in an user-specified target language.

We address the task by using two different approaches that are able to cope with multiple languages: automatic machine translation (MT) and cross-lingual alignment. Automatic machine translation involves translating the news articles into the target language and then applying existing TLS models. Cross-lingual alignment, on the other hand, is based on the idea of representing documents in a language-independent space, where documents in different languages can be compared with each other [154, 18]. In the latter case, the summarization task relies on specific summarization models that leverage aligned embedding representations to select the most salient sentences.

5.3.1 CL-TLS pipeline

The input of the pipeline consists of a set of documents, in multiple languages, that cover a specific event or topic, whereas the output consists of date summaries written in a target language. The proposed methodology consists of three different modules: date selection, summarization and machine translation. We present three alternatives

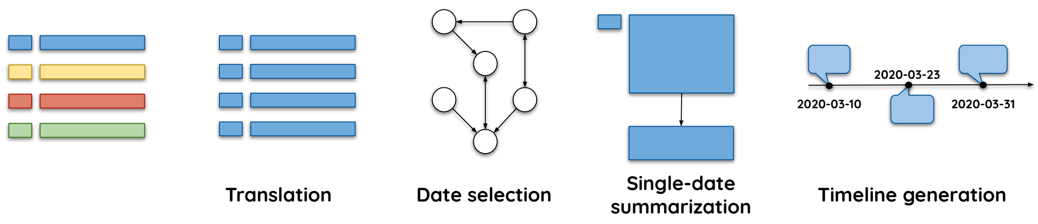


Fig. 5.4 Early translation, CL-TLS pipeline.

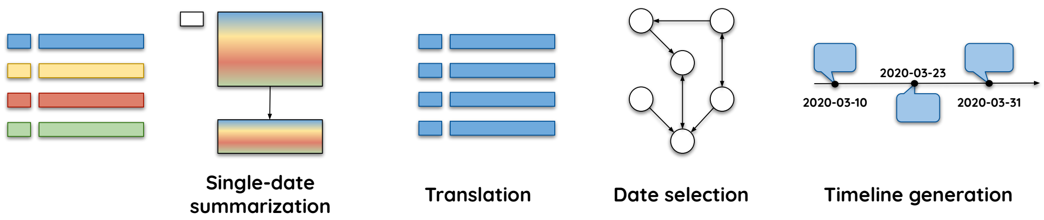


Fig. 5.5 Mid translation, CL-TLS pipeline.

that combine automatic machine translation, cross-language alignment, and standard TLS modules, namely *early*, *mid* and *late*.

- *Early:* it is the most intuitive approach and incorporates the translation step up front in the process. Figure 5.4 shows an overview of the pipeline. The translation of the input documents is followed by the date selection and summarization steps, essentially as in a standard TLS pipeline.

- *Mid*: the cross-language alignment facilitates the repositioning of the summarization step before translation. This approach is shown in Figure 5.5. Before proceeding with the translation step, for each date, we apply language-specific summarization models that summarize the text of each date, separately per language. After the summarization step, we proceed with the translation of

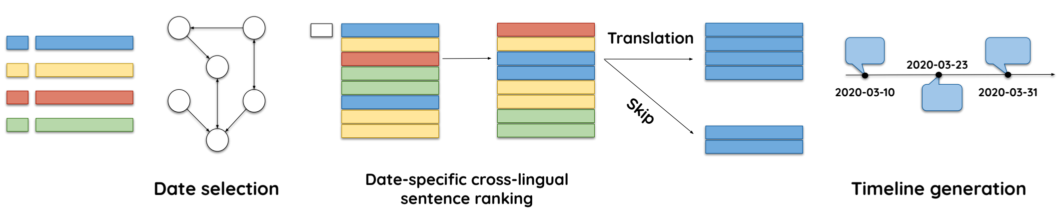


Fig. 5.6 Late/Skip translation, CL-TLS pipeline.

the sentences extracted by the summarizers. The final step select salient dates using graph-based approaches.

- *Late*: it is the most challenging of the three approaches and aims at pushing forward the translation step after the date selection and summarization stages. Figure 5.6 shows the details of this approach. Date selection is the first step in the pipeline. After that, the summarization step uses a set of cross-lingual models to rank sentences according to their estimated relevance. The final date summary can be obtained using either of the following methods:
  - *Translation*: top-scored sentences are translated in the target language, if needed.
  - *Skip*: the ranked list is filtered retaining only the sentences in the target language. The top-scored sentences are then selected to form the final summary. In this case, there is no need for a translation step.

  At the end of the summarization step, the final timeline is generated, similarly to other approaches.

By fixing both the machine translation as well as the date selection models, we independently evaluate the proposed pipelines. Accordingly, we use the mBART model [103] to translate sentences in the target language, and graph-ranking techniques to select the dates (in light of the results discussed in the Section 5.2, we use the in-degree of a node as ranking score).

5.3.2 Multilingual data collection

The cross-lingual timeline summarization task (CL-TLS) requires two different types of data: the first one is a collection of news articles covering the same topic, written in a set of different languages; the second one is a collection of reference timelines, written in a specific target language. Existing datasets for the TLS task are not suitable to address the CL-TLS task, as they contain English-only news articles and English-only reference timelines. Consequently, we propose a novel multilingual dataset called ML-Crisis (*M*ulti*L*ingual-Crisis). It extends the existing Crisis [180] dataset by including three new languages, in addition to English: Italian, Spanish and French. Each language covers four topics, each with its own ground-truth timeline in the target language.

Table 5.4 ML-Crisis dataset statistics.

Language	Avg. # articles	Avg. # sentences	Avg. # timeline dates
English	5114.25	33801.75	25.75
Spanish	197.75	7135.0	45.25
French	117.25	6369.5	65.75
Italian	102.75	5448.75	96.25

ML-Crisis is generated by crawling GlobalVoices [132] and querying Google News[1]. Both sources contain multilingual news articles.

1. We define manually-generated keywords for each pair of language and topic $t_i, l_j \in \mathscr{T}, \mathscr{L}$. For instance, considering the topic related to the *lybia war* and the Italian language, we use the following keywords: 'bengasi, tripoli, scontri, manifestanti, gheddafi'.
2. Similarly, for each pair, we define a reference time period $r(t_i, l_j)$
3. For each language and topic, we query GlobalVoices to retrieve topic-related articles within the reference period. We discard articles that include fewer than two keywords per topic.
4. Using the date range filter set accodingly to the reference period, we query Google News for multilingual articles. Even in this case, we only include news articles that contains at least two topic-specific keywords.
5. We manually collect language-specific timelines for each topic from publicly available resources. Professional journalists from global news agencies typically use standardized and relatively more formal language to report events. CL-TLS systems are evaluated using the collected timelines.

ML-Crisis convers 4 different topics and the same number of languages. Each topic is associated with a list of keywords (also defined in the corresponding languages), a large set of articles in multiple languages, a reference period, and a reference timeline. Dataset statistics are reported in Table 5.4.

[1] https://news.google.com

5.3.3 Metrics for evaluating cross-lingual timeline summarization models

The evaluation of the CL-TLS task should consider the ability of the model to both generate relevant summaries in the target language and to select relevant dates. The metrics for evaluating standard timeline summarization models do not measure supplemental information provided by multilingual enrichment. In order to evaluate the performance of the CL-TLS task, we propose two additional metrics:

- *Enrichment-based (ECL-ROUGE):* it aims at measuring the added value of including other languages in the TLS process. It focuses on comparing the date summaries for dates *missing* in the source articles of the target language. Given a target language L_i and a topic t, we remove from the reference timeline of L_i all the dates that are found in the news articles of the target language. This aims at measuring the enrichment provided by other languages. We compare the modified reference timeline with the system output and compute standard TLS scores. For this metric, single-language TLS systems can not get non-zero scores by construction.
- *Focused-based (FCL-ROUGE)*: It evaluates whether the model can improve content selection by incorporating information available in additional languages. Contrary to ECL-ROUGE, it focuses on dates that are only available in news articles published in the target language. In this case, we modify the reference timeline by keeping only those dates that are available in the target language. Similarly to the other metric, the scores are computed by comparing the modified reference timeline with the system output.

For both metrics, we use specific scores defined for the standard TLS task (see Section 5.2.4).

5.3.4 Experimental evaluation

The validation of the proposed CL-TLS models is conducted using the novel ML-Crisis dataset. All the tests were run leveraging the news articles from the four languages available in the dataset (English, Italian, French, and Spanish). For the unsupervised summarization phase, we use a subset of the summarizers discussed

Table 5.5 Contribution of additional language for date-level enrichment(%).

	Additional Language			
Source Language	English	Spanish	French	Italian
English	0.0	0.02	0.04	0.02
Spanish	0.18	0.0	0.07	0.05
French	0.40	0.19	0.0	0.16
Italian	0.42	0.26	0.15	0.0

in Section 5.2.3. For the *late* configuration, we need to estimate sentence relevance using text in different languages. To this end, we compute sentence relevance using embedding-based summarizers that exploit multilingual sentence embedding models [154].

Multilingual contribution The contribution of multiple languages in the CL-TLS task is crucial. We quantify the contribution of each language to the ML-Crisis dataset by counting the number of dates from the reference timeline that were added by each additional language. When considering language pairs, we report in Table 5.5 the percentage of salient dates added by additional languages. Italian is the language that most benefits from additional information; English news articles reveal 42% of the dates in the reference timeline that were not included in the Italian articles. Unsurprisingly, English is the language that least benefits from additional information, having the lowest contribution from other languages. French and Spanish benefit from extra information provided by other languages, with the former being more influenced by other languages than the latter.

Experimental results - standard TLS metrics Each language available in ML-Crisis is separately set as target language for evaluation. The input of the CL-TLS system is the set of multilingual news articles in all languages while the system timeline is generated in the target language. We test all the pipeline variants and compare them with the single-language equivalent. Using the input documents in the target language is the only option for the latter case.

We report the performance of several unsupervised approaches for the summarization phase. Specifically, we use ELSA [15], TextRank-BM25 [5], and CoreRank [178] as standard summarization model and, similarly to our previous contribution, we evaluate embedding-based versions of SubModular [99], Centroid-Opt [50],

Table 5.6 TLS evaluation for the *Italian* Language. All results are reported in terms of F1-score and highest scores for each metric are reported in boldface.

Summarizer	Date F1	Concat		Agreement		Alignment	
		R1	R2	R1	R2	R1	R2
			Single				
ELSA	0.2535	0.3517	0.0779	0.0296	0.0041	0.0441	0.0051
TextRank-BM25	0.2535	0.3496	0.0794	0.0276	0.0041	0.0420	0.0051
CoreRank	0.2535	0.3253	0.0742	0.0282	0.0042	0.0414	0.0052
EmbeddingRank	0.2535	0.3587	0.0810	0.0298	0.0045	0.0444	0.0055
Centroid-opt	0.2535	0.3502	0.0771	0.0272	0.0037	0.0417	0.0049
Submodular	0.2535	0.3511	0.0781	0.0269	0.0037	0.0410	0.0048
			Early				
ELSA	0.2688	0.4855	0.1567	0.0591	0.0076	0.0766	0.0091
TextRank-BM25	0.2688	0.4816	**0.1659**	0.0651	0.0108	0.0849	0.0131
CoreRank	0.2688	0.5043	0.1652	0.0604	0.0090	0.0806	0.0113
EmbeddingRank	0.2688	0.4550	0.1545	0.0586	0.0078	0.0763	0.0095
Centroid-opt	0.2688	0.4783	0.1615	0.0595	0.0078	0.0778	0.0095
Submodular	0.2688	0.4657	0.1584	0.0594	0.0080	0.0767	0.0097
			Mid				
ELSA	0.2929	0.4851	0.1571	0.0609	0.0073	0.0802	0.0092
TextRank-BM25	0.3113	0.4818	0.1651	0.0706	**0.0122**	0.0898	**0.0143**
CoreRank	0.3133	**0.5211**	0.1636	0.0628	0.0095	0.0857	0.0116
EmbeddingRank	**0.3198**	0.4586	0.1585	0.0713	0.0109	0.0882	0.0130
Centroid-opt	0.3114	0.4856	0.1632	**0.0726**	0.0110	**0.0919**	0.0129
Submodular	0.3002	0.4745	0.1608	0.0660	0.0104	0.0855	0.0125
			Late Translate				
EmbeddingRank	0.2688	0.5062	0.1597	0.0590	0.0083	0.0760	0.0100
Centroid-opt	0.2688	0.5056	0.1601	0.0591	0.0083	0.0760	0.0100
Submodular	0.2688	0.5065	0.1602	0.0591	0.0083	0.0761	0.0100
			Late Skip				
EmbeddingRank	0.2688	0.3093	0.0757	0.0338	0.0058	0.0427	0.0064
Centroid-opt	0.2688	0.3078	0.0755	0.0336	0.0058	0.0427	0.0064
Submodular	0.2688	0.3104	0.0759	0.0337	0.0058	0.0427	0.0064

Table 5.7 TLS evaluation for the *French* Language. All results are reported in terms of F1-score and highest scores for each metric are reported in boldface.

Summarizer	Date F1	Concat		Agreement		Alignment	
		R1	R2	R1	R2	R1	R2
Single							
ELSA	0.2999	0.4235	0.1222	0.0456	0.0086	0.0654	0.0114
TextRank-BM25	0.2999	0.4315	0.1299	0.0505	0.0112	0.0714	0.0144
CoreRank	0.2999	0.4259	0.1233	0.0440	0.0077	0.0636	0.0102
EmbeddingRank	0.2999	0.4108	0.1224	0.0432	0.0082	0.0621	0.0106
Centroid-opt	0.2999	0.4242	0.1233	0.0444	0.0083	0.0640	0.0107
Submodular	0.2999	0.4186	0.1228	0.0453	0.0096	0.0648	0.0122
Early							
ELSA	**0.3446**	0.4317	0.1423	0.0709	0.0089	0.0989	0.0117
TextRank-BM25	**0.3446**	0.4070	0.1515	**0.0823**	**0.0172**	**0.1143**	**0.0219**
CoreRank	**0.3446**	0.4468	0.1500	0.0773	0.0134	0.1075	0.0172
EmbeddingRank	**0.3446**	0.3767	0.1366	0.0708	0.0092	0.0999	0.0132
Centroid-opt	**0.3446**	0.3945	0.1406	0.0721	0.0102	0.1011	0.0137
Submodular	**0.3446**	0.3811	0.1353	0.0713	0.0095	0.1003	0.0131
Mid							
ELSA	0.2651	0.4226	0.1399	0.0540	0.0077	0.0788	0.0101
TextRank-BM25	0.2694	0.4229	**0.1556**	0.0674	0.0156	0.0971	0.0199
CoreRank	0.2837	**0.4479**	0.1452	0.0657	0.0127	0.0927	0.0163
EmbeddingRank	0.2430	0.3833	0.1415	0.0540	0.0113	0.0829	0.0153
Centroid-opt	0.2344	0.4057	0.1419	0.0528	0.0096	0.0816	0.0131
Submodular	0.2535	0.3945	0.1423	0.0558	0.0107	0.0829	0.0142
Late Translate							
EmbeddingRank	**0.3446**	0.4346	0.1333	0.0715	0.0093	0.0991	0.0122
Centroid-opt	**0.3446**	0.4365	0.1337	0.0714	0.0091	0.0993	0.0120
Submodular	**0.3446**	0.4350	0.1334	0.0715	0.0092	0.0991	0.0121
Late Skip							
EmbeddingRank	**0.3446**	0.3896	0.0990	0.0540	0.0080	0.0721	0.0099
Centroid-opt	**0.3446**	0.3917	0.0996	0.0540	0.0080	0.0720	0.0099
Submodular	**0.3446**	0.3915	0.0997	0.0540	0.0080	0.0720	0.0099

Table 5.8 TLS evaluation for the *Spanish* Language. All results are reported in terms of F1-score and highest scores for each metric are reported in boldface.

Summarizer	Date F1	Concat		Agreement		Alignment	
		R1	R2	R1	R2	R1	R2
Single							
ELSA	**0.3478**	0.4104	0.1394	0.0849	0.0302	0.1010	0.0325
TextRank-BM25	**0.3478**	0.4114	**0.1441**	**0.0935**	**0.0338**	**0.1118**	**0.0374**
CoreRank	**0.3478**	0.4242	0.1383	0.0821	0.0264	0.0998	0.0294
EmbeddingRank	**0.3478**	0.4002	0.1332	0.0873	0.0321	0.1038	0.0345
Centroid-opt	**0.3478**	0.4049	0.1318	0.0875	0.0277	0.1040	0.0300
Submodular	**0.3478**	0.3934	0.1352	0.0845	0.0314	0.1011	0.0339
Early							
ELSA	0.3063	0.3108	0.1030	0.0564	0.0097	0.0708	0.0115
TextRank-BM25	0.3063	0.3345	0.1136	0.0669	0.0149	0.0860	0.0182
CoreRank	0.3063	0.3627	0.1125	0.0596	0.0097	0.0775	0.0125
EmbeddingRank	0.3063	0.3068	0.1072	0.0565	0.0089	0.0726	0.0112
Centroid-opt	0.3063	0.3093	0.1034	0.0580	0.0090	0.0742	0.0115
Submodular	0.3063	0.3089	0.1089	0.0577	0.0095	0.0742	0.0121
Mid							
ELSA	0.2659	0.3356	0.1180	0.0547	0.0196	0.0752	0.0218
TextRank-BM25	0.2467	0.3577	0.1307	0.0579	0.0209	0.0796	0.0239
CoreRank	0.2561	0.3390	0.1130	0.0479	0.0134	0.0711	0.0160
EmbeddingRank	0.2806	0.3125	0.1131	0.0564	0.0142	0.0773	0.0171
Centroid-opt	0.2567	0.3146	0.1112	0.0445	0.0081	0.0657	0.0107
Submodular	0.2640	0.3098	0.1086	0.0529	0.0129	0.0726	0.0157
Late Translate							
EmbeddingRank	0.3063	0.3665	0.1094	0.0581	0.0083	0.0732	0.0099
Centroid-opt	0.3063	0.3670	0.1096	0.0582	0.0084	0.0731	0.0099
Submodular	0.3063	0.3668	0.1095	0.0584	0.0083	0.0735	0.0099
Late Skip							
EmbeddingRank	0.3063	0.4301	0.1292	0.0779	0.0234	0.0936	0.0256
Centroid-opt	0.3063	**0.4319**	0.1310	0.0792	0.0251	0.0950	0.0273
Submodular	0.3063	0.4306	0.1292	0.0779	0.0235	0.0936	0.0256

Table 5.9 TLS evaluation for the *English* Language. All results are reported in terms of F1-score and highest scores for each metric are reported in boldface.

Summarizer	Date F1	Concat		Agreement		Alignment	
		R1	R2	R1	R2	R1	R2
				Single			
ELSA	0.2254	0.3018	0.0460	0.0323	0.0037	0.0420	0.0044
TextRank-BM25	0.2254	0.3238	0.0568	**0.0413**	0.0075	0.0522	0.0088
CoreRank	0.2254	0.2994	**0.0575**	0.0362	0.0070	0.0490	0.0090
EmbeddingRank	0.2254	0.2869	0.0481	0.0369	0.0051	0.0480	0.0064
Centroid-opt	0.2254	0.3079	0.0521	0.0373	0.0053	0.0491	0.0069
Submodular	0.2254	0.2942	0.0508	0.0369	0.0053	0.0486	0.0066
				Early			
ELSA	**0.2298**	0.2807	0.0374	0.0285	0.0049	0.0357	0.0054
TextRank-BM25	**0.2298**	0.3213	0.0516	0.0410	0.0077	**0.0538**	0.0092
CoreRank	**0.2298**	0.2823	0.0549	0.0387	**0.0090**	0.0499	0.0107
EmbeddingRank	**0.2298**	0.2868	0.0485	0.0347	0.0040	0.0453	0.0048
Centroid-opt	**0.2298**	0.2955	0.0481	0.0369	0.0050	0.0470	0.0063
Submodular	**0.2298**	0.2889	0.0496	0.0344	0.0054	0.0454	0.0068
				Mid			
ELSA	0.1829	0.3003	0.0462	0.0205	0.0026	0.0320	0.0048
TextRank-BM25	0.2082	**0.3246**	0.0570	0.0357	0.0083	0.0478	**0.0112**
CoreRank	0.1568	0.2926	0.0483	0.0227	0.0047	0.0360	0.0081
EmbeddingRank	0.1707	0.2960	0.0451	0.0253	0.0050	0.0406	0.0073
Centroid-opt	0.1445	0.3063	0.0440	0.0201	0.0041	0.0314	0.0059
Submodular	0.1637	0.2985	0.0456	0.0204	0.0038	0.0348	0.0058
				Late Translate			
EmbeddingRank	**0.2298**	0.2915	0.0427	0.0222	0.0037	0.0317	0.0047
Centroid-opt	**0.2298**	0.2931	0.0423	0.0221	0.0037	0.0317	0.0047
Submodular	**0.2298**	0.2920	0.0428	0.0223	0.0038	0.0319	0.0048
				Late Skip			
EmbeddingRank	**0.2298**	0.2839	0.0397	0.0236	0.0041	0.0321	0.0046
Centroid-opt	**0.2298**	0.2842	0.0396	0.0234	0.0041	0.0319	0.0046
Submodular	**0.2298**	0.2842	0.0398	0.0237	0.0041	0.0322	0.0046

Table 5.10 Evaluation of CL-TLS metrics. All scores are reported as F1-scores.

Target language	Date F1 ECL	Date F1 FCL	Concat R2 ECL	Concat R2 FCL	Agreement R2 ECL	Agreement R2 FCL	Alignment R2 ECL	Alignment R2 FCL
English	0.0	0.2298	0.0	0.0337	0.0	0.0049	0.0	0.0054
Spanish	0.0526	0.3440	0.0604	0.1008	0.0148	0.0226	0.0181	0.0240
French	0.2007	0.3869	0.1082	0.1088	0.0112	0.0187	0.0158	0.0219
Italian	0.1802	0.3648	0.1369	0.1070	0.0072	0.0133	0.0092	0.0147

EmbeddingRank [138] that leverage multilingual sentence embeddings [154] to estimate sentence ranking (see Section 5.2.3).

Tables 5.6, 5.7, 5.8, and 5.9 reports the results obtained when setting the target language to Italian, French, Spanish and English, respectively. Unsurprisingly, Spanish and English are the two languages that least benefit from the support of other languages. Since those two languages are the least affected by the inclusion of other languages in the reference timeline, we observed, on average, a lower level of improvement that was obtained in all multilingual settings. Both Italian and French, on the other hand, are the languages that most benefit from the inclusion of other languages in the reference timeline. The mid-translation pipeline shows better performance than all the others for Italian while early-translation improves the performance for French considering *agreement* and *alignment* metrics. For those languages, we also run statistical significance tests [136] comparing the performance of the early-translation pipeline with the other approaches. By setting the significance level to $p = 0.05$, we compare the results obtained by embedding-based summarizers (i.e., those available across all pipelines). We observe that the results obtained using *early* and *mid* pipelines are statistically superior to the others considering the ROUGE-1 alignment metric. Neither the *late-translate* nor *late-skip* pipelines perform significantly better in any of the four target languages, recommending further investigation of these approaches.

Experimental results - CL-TLS metrics To evaluate the impact of cross-lingual enrichment for timeline generation, we conduct an analysis according to the evaluation criteria discussed in Section 5.3.4. Leveraging the *early-translation* pipeline and TextRank-BM25 summarizer for all target languages, we report the *FCL* and *ECL* ROUGE-2 scores in Table 5.10. Given the limited contribution of other languages to English (see Table 5.5), its ECL scores never exceed 0. The results for Italian and

French confirm the previous findings regarding the benefit of the cross-lingual enrichment, with high ECL-scores both for *date selection* and *concat* metrics. Considering FCL scores, the English language shows very limited benefits from the cross-lingual enrichment, whereas all other languages gain a significant amount of information from the multilingual data. This results confirm that the English language already contains enough information to be used for the task, whereas for other languages the cross-lingual enrichment plays a significant role.

5.4 Challenges and future works

This chapter provided an overview of the temporal aspects of the summarization task, focusing on timeline summarization both from single and multilingual perspective. Despite its unique characteristics and challenges, this task is still under-explored, especially in multilingual settings. The main goals of the task are the identification of the most relevant dates and the selection of the most salient information. When considering multiple source language, cross-lingual information retrieval and enrichment are crucial to identify the most relevant terms and entities.

Timeline summarization research has focused mainly on unsupervised models, however, the cross-lingual scenario highlighted the benefit of additional information sources. The inherent complexity of the task and the variety of events that can be represented in a timeline, require the development of specific models and features. The lack of annotated datasets is a major challenge for the development of effective and accurate methods for timeline summarization, expecially in multilingual settings. In the current context, datasets are limited in term of size and the kind of information that can be represented.

The recently proposed TLS-Covid19 [140] and Timeline-100 [95] datasets can help the community in this direction. The former is an English-Portuguese dataset with annotated information about the COVID-19 pandemic. It covers 178 sub-topics and contains more than 100,000 news articles, thus providing a comprehensive overview of the events that have occurred since the outbreak of the pandemic. The availability of multiple languages for this dataset may also help the community in analyzing the multilingual challenges of the task. Timeline-100 is a multi-topic dataset covering different fields (e.g., economy, military, and education among others). It includes more than 10,000 English news, as well as 100 manually-annotated reference timelines. The release of new datasets and the improvement of current methods can also help with the development of effective learning models.

The majority of the existing datasets for timeline summarization are text-based. This task, however, can be very useful in guiding automated content curation from large multimedia archives. In this perspective, the task can benefit by the inclusion of multimodal information, such as images and videos, to create a better understanding of the events and their relations with other events and entities. The generation of the topic timeline can also be enhanced by the use of computer vision models to automatically identify and label the entities and events from images and videos. This scenario may require the use of recent multimodal models. For example, the recent CLIP model [148] provides cross-modal embeddings for natural language and images; the application of this approach to the task of timeline summarization could be an interesting research direction.

Another open question to be explored in the future is the integration of multilingual information in the timeline generation. It was shown in this chapter that single-language approaches could be adapted to solve this problem, but its performance is still far from the state-of-the-art results achieved on English. The CL-TLS task is, however, very useful when multilingual information is available, as the identification of the most relevant dates and information in multiple languages can help to understand the process of events unfolding across the globe. Consequently, future research may focus on cross-lingual entity linking and cross-lingual event extraction to overcome the current issues of the task and propose more effective methods to generate multilingual timelines.

Most existing approaches for the TLS task are based on extractive summarization. However, the task can also be addressed from the perspective of abstractive generation [170, 25], which is more challenging but can generate more fluent summaries. The abstractive approach has the advantage of overcoming the limitation of the extractive approach (i.e., it can only select sentences from source articles). In this direction, the use of sequence-to-sequence models can help to improve the current performance of the task. The use of abstractive summarizers, however, is not straigthforward. They may require the design of specific strategies to ensure that the correct ordering of the events is considered in the summary and to prevent the generation of false information [195].

Finally, timeline summarization can be improved with the introduction of new evaluation metrics, such as evaluating the temporal awareness of systems. In this chapter, the timeline generation task was assessed exclusively using automatic evaluation metrics, but this approach does not account for the temporal coherence of the timeline. This aspect is especially relevant for multilingual summarization. The events can be described differently across languages, and creating a coherent timeline can be challenging. The inclusion of human evaluation as further assessment can help to better validate the temporal coherence of the generated timeline.

Chapter 6

Understanding and summarizing spoken language

Natural language understanding is a challenging task that has been extensively studied in computer science, particularly in the text domain. Natural language, however, is not limited to the written form but is also expressed through other modalities such as recorded speeches. The ability to comprehend speech is crucial for artificial intelligence since it is humans' primary mode of communication. This chapter aims at providing some insights into automatic content summarization of spoken language. In Section 6.1 we provide an overview of the research works that focus on spoken content analysis, discussing the main application domains and the most common deep neural network architectures used for this task. Section 6.2 introduces our first contribution to this field by introducing a multimodal approach for the extractive summarization of podcasts. Section 6.3 is dedicated to our second contribution, the participation to the TREC 2021 Podcast Track, where we propose a deep learning-based abstractive summarization approach. Finally, Section 6.4 outlines future research directions that could be explored in this field. Figure 6.1 provides an overview of the contributions discussed in this chapter, as well as the relevant fields of study.

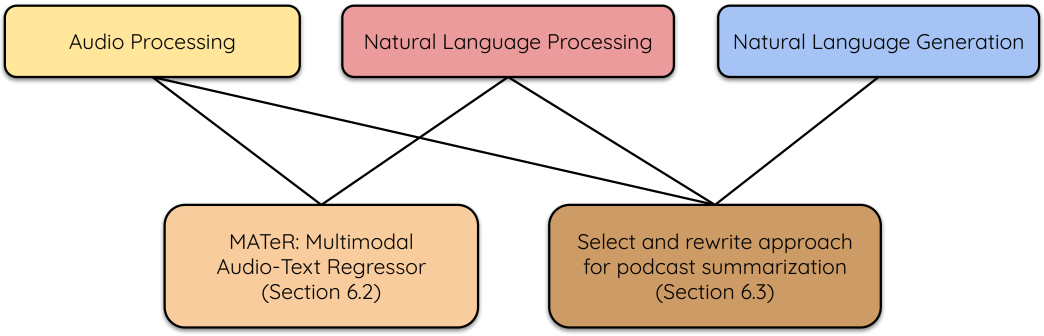


Fig. 6.1 Graphical overview of the topics and contributions covered in Chapter 6

6.1 Related works on spoken content analysis

The analysis of spoken content is a challenging task which has attracted the attention of the research community in the past decades. It has been successfully applied in various research fields, such as automatic speech recognition [24], speaker identification [4], and speech synthesis [201] among others. Traditional approaches to the analysis of spoken content were largely based on handcrafted and statistical features [91, 42], however, this field has considerably benefited from the advent of deep learning architectures, which allow to automatically learn audio representations that are capable of capturing the inner structure of the data. Access to a large amount of digital speech, e.g., in the form of podcasts or call recordings, has also been an essential factor in the development of deep learning models, since they allow data-driven models to be trained using self-supervised learning [3].

6.1.1 Audio representation learning

Within the past decade, the field of text processing has undergone a profound transformation, with deep learning architectures greatly improving performance across various tasks. They are able to learn latent representations that model the inner structure of the data, thus providing a better understanding of their meaning. Incorporating this concept to the analysis of audio sequences would enable the design of models capable of automatically learning representations that capture the semantics of the input signal.

Analyzing audio signals, however, is much more challenging than text analysis. Unlike textual sequences, which are directly related to a finite set of symbols, acoustic sequences are continuous signals that change over time, which makes discretization an essential step for their analysis. Convolutional neural networks (CNN) are the most popular neural architectures to extract high-level features from image data, and they have been successfully applied to audio signals [193]. CNNs can learn to detect patterns in the input signal and extract features related to its semantics. These models have been successfully applied to various tasks, such as acoustic scene classification [6] and sound source separation [21].

The use of self-attention mechanisms in transformer-based architectures has shown impressive performance in the field of NLP. However, they operate on discrete symbols (e.g., words or tokens) so they can not be directly applied to acoustic signals. Wav2Vec 2.0 [3] is a transformer-based architecture designed for the task of Automatic Speech Recognition (ASR). It relies on a quantization step that converts raw speech into a discrete sequence of symbols, which can then be processed by the transformer model. By masking some parts of the input signal, the model is trained to predict the correct quantized representation of the signal at the masked positions. Similarly to transformer models for NLP, signal masking allows the model to learn global patterns and long-term dependencies in the input signal. In these settings, a model can be trained on audio signals without having to rely on transcripts or any other type of supervision. Wav2Vec 2.0 is primarily designed to learn feature representations for ASR tasks, however, these representations may be useful for a wide range of tasks in the audio domain as well. The original architecture has also been extended by including clustering techniques during the feature-extraction phase [65] or denoising pre-training objectives [24].

Transformer models have been shown to have potential for learning useful feature representations in the audio domain. However, they tend to be too geared toward ASR and speech-related tasks [190], thus potentially lacking the ability to learn general feature representations that could be tranferred to other audio-related tasks. The generalization of transformer-based architecture to other audio tasks is an open research area that is currently being explored [54, 55].

6.1.2 Textless NLP

The deep learning revolution in the field of NLP has been characterized by the ability to automatically extract semantic representations from text. Spoken content analysis, offers instead a unique opportunity to learn representations directly from speech signals that are much richer than the text transcriptions. A new research trend in the field of NLP has recently been identified as *textless NLP* [88]. It seeks to learn from speech signals in an end-to-end manner, rather than extracting textual features through automatic speech recognition. ASR, which is an essential part of traditional spoken content analysis, might not be needed for many applications, including speaker identification, speaker diarization, and emotion recognition. Additionally, it may hinder effective feature learning, since (i) the automatic transcription might not be accurate enough for the task at hand, and (ii) only a small portion of the information in speech can be captured in text.

Generative models have been used to tackle various speech-related tasks without ASR. Speech emotion conversion, for example, can be formulated as a domain-adaptation problem, in which the goal is to learn to convert speech signals from one emotion to another [81]. Textless models are able to generate emotion-converted speech without ASR, thus removing the need for text transcriptions or alignments. Similar models have been proposed for speech-to-speech machine translation [68], natural dialogue generation [134], and speech resynthesis [143]. Textless NLP has been gaining considerable research interest and popularity in recent years. In Section 6.4.2 we identify various research opportunities in this field and the challenges that need to be addressed.

6.1.3 Spoken content summarization

Most of the contributions discussed in this thesis pertain to text summarization, which involves generating a shorter, more concise version of a document without altering its original content. The notion of summarization can be applied to spoken content as well. Therefore, the task of summarizing spoken content becomes the problem of extracting or generating a summary from an audio file. A summary can be either spoken (audio) or written (text), and it should be concise, accurate, and include most of the information presented in the original audio snippet.

Most spoken summarization approaches involves a two-steps process: first, automatic speech recognition (ASR) converts the audio waveform into text; then, the summarization system generates the summary from the transcribed text. Those approaches have the advantage of being generic and applicable to any spoken content, however, the automatic speech recognition step may introduce transcription errors, which are propagated to the summarization step. To mitigate the error propagation problem, Li et al. [93] propose a hierarchical ASR and summarization system aiming to minimize errors by combining semantic segmentation and merging algorithms. Similarly, the abstractive summarization of meeting recordings has been addressed by using hierarchical models that impose diversity at inference time by modeling utterance-level attention distribution during training [107].

Two-stage approaches, however, remain limited by the automatic speech recognition component and may not be ideal for specific domains where the speech recognition process is challenging. An alternative approach is to develop systems that work directly on the audio waveform. These models can be directly applied to audio snippets without the need for transcription, which makes them more robust to errors in automatic speech recognition.

Recent studies have shown that combining acoustic features with traditional summarization objectives provides a promising solution for summarizing spoken content [102]. Most of these approaches leverage transformer-based models to learn acoustic features from the raw audio waveform in an end-to-end manner. Their main drawback is the high computational cost, which can be problematic when scaling to large data sets, especially when dealing with long audio recordings. Recent proposals of efficient architectures that harness restricted-attention mechanisms can reduce computational complexity and generate insightful summaries of spoken audio content [166].

6.2 MATeR: Multimodal Audio-Text Regressor

Audio podcasts have gained increasing popularity as a way to consume audio content and are widely used for entertainment and educational purposes. They provide a convenient way for a user to consume audio content and are often long-form, which makes it challenging for users to identify their topic and decide whether they are worth listening to. Our contribution to the field is to devise methodologies for

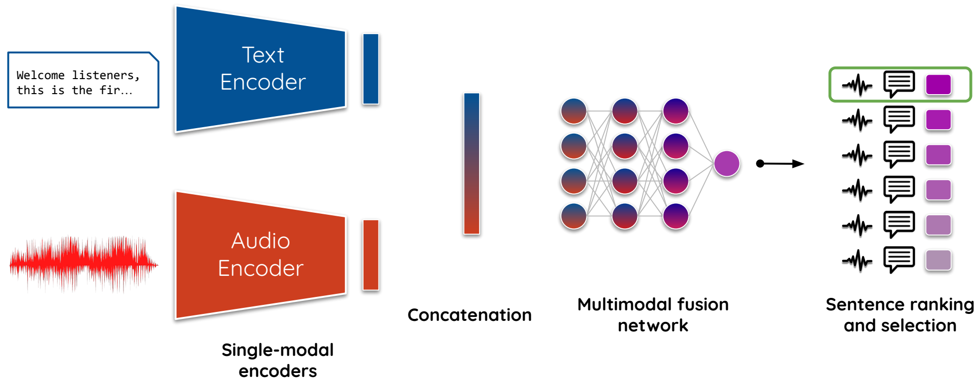


Fig. 6.2 MATeR's architecture sketch.

automatically generating short summaries of podcast episodes, thereby providing users with a quick overview of the podcast content without having to listen to the entire podcast.

Podcasts typically contain spoken content, and thus an automatic summarization of podcasts can be performed using the two-stage approach outlined in Section 6.1.3. The first stage is to transcribe the spoken content of the podcast using ASR, and the second stage is to produce a summary of the transcribed text [186]. Although this approach to podcast summarization is straightforward, it has several limitations. First, ASR results are often error-prone, and therefore the transcript may be incorrect, which can adversely affect the summarization results. Second, considering only the transcript in the summarization process can lead to suboptimal results, as the spoken content incorporates other important cues, such as intonation, which are not conveyed in the transcript. Hence, it would be beneficial to consider audio data in the podcast summarization process to complement the transcribed text.

To address the aforementioned limitations, we propose a multimodal architecture that incorporates both the acoustic features and the transcribed text of a podcast, namely MATeR (Multimodal Audio-Text Regressor) [185]. The system leverages an end-to-end deep learning architecture to predict the relevance score of each sentence in the transcript, and then generates a summary by selecting the most relevant sentences. MATeR architecture is depicted in Figure 6.2. It includes three main components: (i) an acoustic encoder that extracts features from the podcast recording; (ii) a text encoder that extracts features from the transcript; and (iii) a multimodal fusion network that combines the acoustic and text features to predict the relevance score of each sentence.

Text Encoder The text encoder is designed to extract robust representations of sentences included in the podcast transcripts. The transformer architecture [187] can provide effective encoding strategies across a variety of NLP tasks. Pre-training transformer models with large corpora enable efficient transfer learning for downstream tasks, supporting the fine-tuning of powerful text encoders even with limited training data. In our work, we use BERT tranformer model [38] pre-trained on English language corpus.

The encoder model takes as input a sequence of text tokens corresponding to each sentence in the transcript. The tokenized sequence is mapped to token embeddings, which are then passed to a transformer-encoder stack. Our model needs to generate fixed-length representations of variable-length input sequences, thus we leverage on a special `[CLS]` token inserted at the start of the sequence. The embedding vector corresponding to `[CLS]` is taken as the sentence embedding for the whole sequence [153]. Each sentence is represented by a 768-dimensional vector which encodes information from the entire input sequence.

Audio Encoder The audio encoder aims to extract features from the audio waveform and aims at capturing the contextual structure of the speech signal. It implements the Wav2Vec 2.0 model [3], which, similarly to the text counterpart, is a self-supervised model trained on large amounts of speech data. The audio encoder includes a feature extraction step followed by a stack of transformer layers. The feature extractor leverages a convolutional neural network to obtain frame-level representations (i.e., audio tokens), which are then passed to the transformer layers. This results in a sequence of audio tokens that are processed by the transformer stack to learn contextualized representations that capture both local and global dependencies in the audio waveform.

The model, similarly to the text encoder, generates a sequence of contextualized audio tokens, whereas a single representation is needed for the entire waveform. Thus, contextualized tokens are passed through an average pooling layer to obtain a single audio vector (i.e., the audio encoder output) that represents the entire input sequence as a 768-dimensional vector.

Multimodal Fusion Network Audio and text encodings are combined to produce multimodal sentence representation. Each modality-specific module encodes different information, which can be exploited to model the relationship between the two modalities. Text and audio embeddings are first concatenated to form the joint embedding vector, which is then passed to the multimodal fusion network for further processing. The fusion module consists of a set of fully-connected layers, which are trained to predict a relevance score r_s given the joint embedding vector. The relevance score vector $r_s \in \mathbb{R}$ is a real-valued vector in the range $[0, 1]$ and indicates the importance of the audio-text pair in the podcast episode.

In our model, the fusion network comprises three fully-connected layers, each having a width of 1536 and a rectified linear unit (ReLU) activation function [127]. The relevance score for multimodal input is calculated by applying a sigmoid function to the output of the third fully-connected layer. During training we use the Mean Squared Error (MSE) loss to minimize the difference between the predicted relevance score for the training examples and the ground truth relevance score. The loss is back-propagated through the multimodal fusion network and uni-modal encoders to update parameters of the entire model.

6.2.1 Data collection and labeling process

The training and validation of MATeR require a dataset containing the audio of the podcast, its transcription and a ground-truth summary that can be used to train the model. The *Spotify Podcast Dataset* [28] was used for this purpose. It includes $100,000$ podcast episodes spanning over $18,000$ different shows, with their audio files, transcriptions and metadata. The dataset does not contain any specific summary of the podcast content, but it does provide the author's description of the podcast episodes. This description, which is typically short, was used as the ground-truth summary of the podcast content. A random sample of 10% of the episodes in the dataset was used to generate the train set while the test set consisted of 1% of the data.

The transcriptions in the data collections are partitioned into sentences and annotated with their corresponding start and end times to obtain aligned audio-text pairs (e.g., the input of MATeR). Given a text-audio pair as input, our system predicts its relevance score within the podcast. Table 6.1 provides statistics of the data

Table 6.1 Statistics of the podcast dataset used for training and testing MATeR model.

	Episode per Show		Sentence per Episode		Words per Sentence		Words per Description	
	Avg #	Max #	Avg #	Max #	Avg #	Max #	Avg #	Max #
Train	5.77 ± 16.33	351	84.52 ± 56.97	574	72.85 ± 33.34	175	61.56 ± 60.22	709
Test	5.38 ± 13.71	122	78.07 ± 49.91	501	75.99 ± 31.77	183	68.81 ± 54.25	461

collection reporting the average and maximum number of (i) episodes per show, (ii) sentences per episode, (iii) words per sentence, and (iv) words per description.

Podcasts' description cleaning The podcast description is a short summary of the podcast content, which is provided by the content creator and may contain advertising and other unrelated content. For example, the description may include the links to other products from the same publisher, to other episodes, and to the social media accounts of the content creator. In order to prevent bias in the summarization process, we manually label 2200 sentences, pertaining to 400 podcast descriptions, and fine-tune a BERT-based model to predict whether each sentence contains irrelevant content or not. We split the annotated sentences into training and test sets, with a ratio of 80% and 20%, respectively. The resulting data collection contains 39.2% and 35.4% of irrelevant sentences in the training and test sets, respectively.

We capitalize on the contextualization capabilities of BERT by using, for each sentence, the previous one as context and concatenating both sentences with a special token (i.e., `[SEP]`). For the first sentence in the podcast description, we use a special token (i.e., `__START__`) as the context. We fine-tune the binary classification model using Adam optimizer [106] and a constant learning rate $\eta = 10^{-5}$ for 2 epochs. The fine-tuned model achieves an overall accuracy of 0.92 on the test set.

The resulting model is used to label all sentences in the podcast descriptions in our data collection. We remove sentences containing irrelevant content, keeping only the sentences that are relevant to the podcast itself. The automated process acts as an initial filtering step for podcast descriptions and can also be used for other tasks of interest in the podcast domain. Both the manual annotation as well as the fine-tuned model have been made accessible for research purposes[1].

[1] https://github.com/MorenoLaQuatra/MATeR

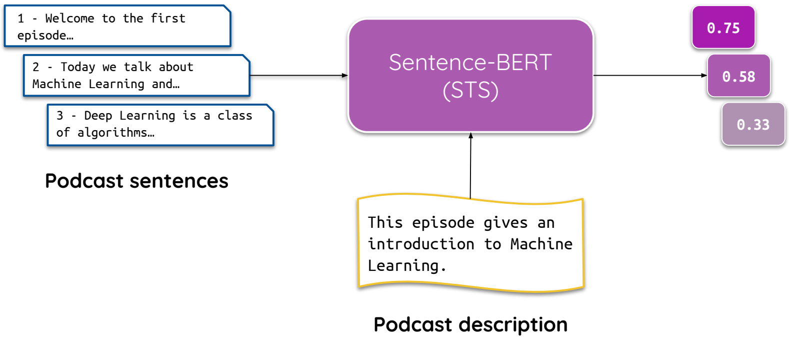


Fig. 6.3 Sketch of the labeling process

Data labeling The relevance score r_s of a sentence in a podcast episode $s \in P$ is represented by a float value in the range $[0,1]$, with 0 meaning that the sentence is not relevant to the podcast description and 1 meaning that the sentence perfectly match the description. Podcast descriptions differ greatly from podcast transcripts in their writing style; while the former is more colloquial and free, the latter is usually written in a structured way, sometimes reporting the podcast topics in a bullet point list. Thus, the use of lexical features to label podcast descriptions would be ineffective and we instead use a semantic approach.

The relevance score is computed according to the semantic similarity of each sentence $s \in P$ with its corresponding podcast description P_d. Specifically, the score is computed by leveraging the cosine similarity between the sentence and the podcast description embeddings. To obtain an accurate estimation of the semantic similarity, we use Sentence-BERT model [153] to encode the sentences and the podcast descriptions. It leverages BERT model and is fine-tuned for the Natural Language Inference (NLI) task. Given a premise and a hypothesis, the model is trained to predict whether the premise entails, contradicts, or neither entails nor contradicts the hypothesis. The NLI task has proven to be helpful for generating semantic-aware sentence embeddings, as the model is required to identify the semantic relationships between premise and hypothesis [34].

Figure 6.3 shows an example of the data labeling process. The podcast sentences, on the left, and the podcast description, on the bottom, are encoded using Sentence-BERT and each sentence vector is separately compared with the description vector. Although only the textual content is considered in the annotation phase, labels are assigned to the audio-text pairs. The final dataset is composed of text, audio pairs coupled with their estimated r_s score, that is used as target label to train the MATeR model.

6.2.2 Experimental evaluation

The proposed architecture is trained to rank podcast's sentences according to their overall relevance to the podcast description. The top-ranked sentence is extracted as the summary of the podcast, and it is compared to the podcast description written by the original author. In order to capture both the relevance of the generated summary and its fluency, semantic and lexical similarity metrics are used to evaluate the quality of the generated summaries. The lexical evaluation is carried out using the well-established ROUGE metric [97], while the semantic evaluation is carried out using BERT-based sentence similarity, which aims to capture the semantic similarity of the generated summary to the original description. Among ROUGE scores, we consider ROUGE-1, ROUGE-2, and ROUGE-L, which are computed based on the overlap between unigrams, bigrams, and longest common subsequences. In order to determine semantic similarity, the embedding of the generated summary is computed using the Sentence-BERT model [153] and compared to the embedding of the podcast's description.

Podcast's transcript are generally very long, containing on average $\sim$ 84 sentences of $\sim$ 73 words each, as reported in Table 6.1. Typically, Transformer-based models have input token limitations of 512 or 1024. MATeR overcomes this issue by processing each sentence in a podcast transcript separately to estimate its relevance score. The length of the text hinders the use of standard supervised extractive summarization models [104, 204] which consider the sequence of sentences in a podcast transcript as a whole.

To evaluate the effectiveness of MATeR, we compared it with several of the standard extractive summarization models that can handle long text sequences.

- *LEAD-1*: the summary is extracted by selecting the first sentence of the podcast's transcript.
- *TextRank [122]*: it is an unsupervised baseline that ranks transcript's sentences according to their relevance score computed using the TextRank algorithm. The top-ranked sentence is selected as the summary.

Table 6.2 Comparison of podcast summarization approaches using Rouge-1, Rouge-2, Rouge-L, and SBERT scores. Highest scores for each metric are reported in boldface and statistically significant performance improvement (p-value= 0.05) are starred.

Method	R1-P	R1-R	R1-F1	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	SBERT
LEAD-1	0.150*	0.170*	0.142*	0.014*	0.013*	0.011*	0.129*	0.147*	0.122*	0.350*
TextRank	0.154*	0.177*	0.147*	0.015*	0.016*	0.013*	0.133*	0.154*	0.127*	0.363*
CoreRank	0.176*	0.179*	0.157*	0.030*	0.024*	0.023*	0.152*	0.154*	0.135*	0.418*
TextRank-BM25	0.156*	0.203*	0.159*	0.017*	0.020*	0.015*	0.132*	0.174*	0.135*	0.414*
HiBERT	0.186*	0.219*	0.184	0.036*	0.033*	0.031*	0.162*	0.191*	0.160	0.482
MATeR-text	0.162*	0.168*	0.143*	0.016*	0.016*	0.013*	0.140*	0.146*	0.123*	0.348*
MATeR	**0.193**	**0.225**	**0.188**	**0.042**	**0.041**	**0.036**	**0.168**	**0.197**	**0.164**	**0.490**

- *TextRank-BM25 [5]*: the summary is extracted according to the TextRank algorithm, with the similarity scores being computed using the BM25 retrieval function [156].

- *CoreRank [178]*: it is an unsupervised baseline that leverages both submodular function optimization and graph-based representations of the podcast's transcript. The CoreRank algorithm provide a set of sentences that are ordered according to their relevance scores. We extract the summary considering the top-ranked sentence in the set.

- *HiBERT [203]*: it is a supervised baseline that relies on hierarchical Transformers to extract the summary. The HiBERT model is fine-tuned using the same training set used to train the MATeR model. It provides a binary label for each sentence in the podcast's transcript, indicating whether or not it should be included in the summary. We use the posterior probability of the predicted label to rank sentences and extract the summary considering the top-ranked one.

- *MATeR-text*: it is the text-only version of the proposed architecture. It is composed of a text encoder and a regressor layer. The model is trained by using the transcript only. The final summary is extracted considering the top-ranked sentence, based on the estimated importance score.

Table 6.2 reports the results obtained by the above-listed models using ROUGE-1 (R1), ROUGE-2 (R2), ROUGE-L (RL), and Sentence-BERT (SBERT) evaluation metrics. For ROUGE-based scores, we include the Precision (P), Recall (R) and F1-score (F1)

Table 6.3 Qualitative comparison of MATeR, HiBERT, and MATeR-text for the extraction of podcasts' summaries. Text including topic's description or a catch phrase is highlighted in green and blue, respectively.

Author's provided description		
Brandon Bozarth is a Spiritual coach and teacher that focuses on the process of integration. Being an avid studier of neuroscience, quantum physics, ancient teachings, consciousness, metaphysics, psychology, behavior design and so on... he progressively has become a leader and catalyst to the what he calls the new earth. That new earth that is being birthed, a world based in love, collaboration, community, and growth. Brandon teaches people how to remember who they are from the higher realms and the history of our existence. He teaches deep spiritual Self empowerment and that miracles are a natural part of our existence. Brandon breaks down the complicated but fascinating history of our galactic universe and is able to contact and channel different beings from other dimensions to communicate information from the higher realms right to us. We talk about starseeds, alien races, our galactic history in relationship to those alien races and a heavy dose of quantum concepts such a inter-dimensionality and the construct of time. This is a juicy one guys!		
MATeR	**HiBERT**	**MATeR-text**
Of our existence he teaches deep spiritual self empowerment and that Miracles are natural part of our existence Brendan breaks down the complicated but fascinating history of our Galactic Universe and is able to contact and channel different beings from other dimensions to communicate the information from the higher Realms straight to us. I'm very excited for you guys to hear this episode. It's going to be a good one.	And even dating some of the shows will be just me rambling about my mystical experiences and new discoveries. In other shows will have guests to open up New Perspectives and Views. I hope you'll join me on this journey as we discuss in open up about what spirituality in today's world really looks like	with so much love. I was just like crying from Love and having this conversation with one of the beans. He was blue. It was kind of like a guide of mine named Grace and who really just represents the state of grace and offers that to me and then the other one was an artery is being which is orange. So they represent themselves as the color orange. Typically there. They have a like a phone number you could say which is a eight-pointed orange star kind of the long star and that's kind of like

As expected, the LEAD-1 approach performs worst. This is because it uses the simplest selection technique, i.e. selecting the first sentence of the podcast transcript, which is usually not an accurate summary of the entire podcast. In contrast, the TextRank, CoreRank, and TextRank-BM25 models perform quite similarly, with CoreRank performing slightly better than the other two. This is probably due to the similarity of these approaches, which are based on similar graph-based selection techniques.

The HiBERT model, which is trained in supervised settings, outperforms all the other methods except for the MATeR model. This is not surprising since HiBERT is fine-tuned to select sentences from podcast transcripts, while the unsupervised models are not. Finally, the MATeR model outperforms all the other methods, both supervised and unsupervised, according to all the evaluation metrics. Furthermore, we report the results of a MATeR-text model that only extracts summary information using text-based features. This model performs significantly worse than MATeR, indicating that audio information is indeed helpful to extract summary sentences.

Qualitative evaluation Table 6.3 shows a qualitative comparison of the three best methods (i.e. MATeR, HiBERT, and MATeR-text) for the extraction of a podcast's summary. For the sake of readability, we highlight sentences according to their intent using different colors.

Both MATeR and HiBERT are able to extract the podcast's summary quite accurately, whereas MATeR-text is not able to extract any sentence that introduces the podcast. Although HiBERT can find a sentence that contains some information about the podcast's topic as well as a catch phrase, MATeR is able to find a sentence that contains more information about the podcast's topic. MATeR's proposal has a higher quality summary since it can leverage both acoustic and textual content, whereas HiBERT only relies on textual content. Speaker's intonation, emotion, and word stress are important elements in speech, which are not captured in the textual content only. More details on the qualitative evaluation as well as the audio samples extracted by each method are provided in the official project repository[2].

6.3 Select and rewrite approach for podcast summarization

The analysis of podcast content is challenging because audio content is unstructured, and transcripts could contain noisy information due to automatic speech recognition (ASR) errors. The podcast track at *Text REtrieval Conference (TREC)* [75] aims at fostering research in various aspects of podcast information access. The 2021 edition of the podcast track comprises two tasks: segment retrieval and podcast summarization. Podcast segment retrieval is a traditional information retrieval task in which, given a query, the goal is to retrieve the most relevant podcast segments of two minutes in length. Podcast summarization, instead, focus on the generation of a single summary for each podcast episode.

Similarly to the MATeR model [185] described in Section 6.2, we address the podcast summarization task by proposing a multimodal strategy that combines the audio and the transcriptions of a podcast. The goals of the proposed task are twofold: (i) to generate a short audio summary, up to 60 seconds, from the podcast, and (ii) to provide a textual summary of the podcast episode that can be displayed in

[2] https://github.com/MorenoLaQuatra/MATeR

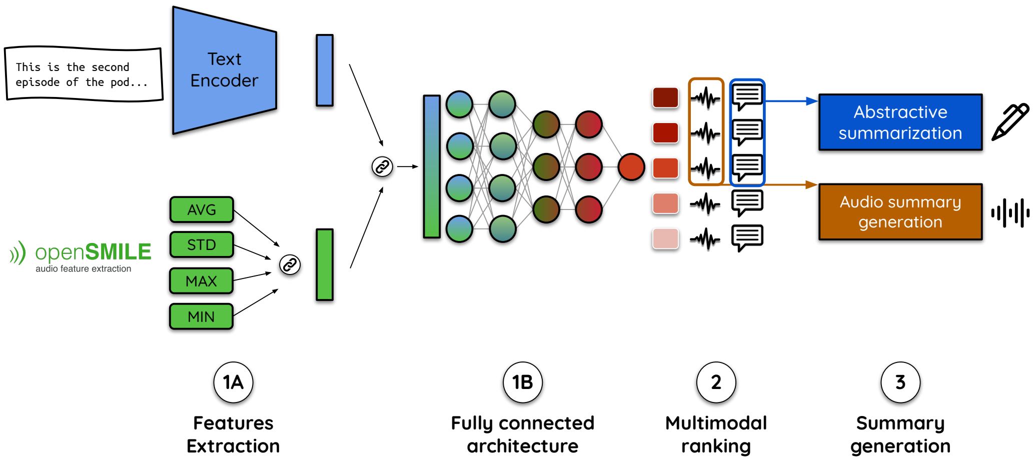


Fig. 6.4 Select and rewrite architecture for podcast summarization.

the interface of a podcast player. Our aim is to generate two summary types that are complementary, so that the listener can choose the summary type that is more suitable according to the context of use and the available time. The proposed strategy combines both audio and text features for the *selection* of the most relevant content of the podcast and then *rewrites* the selected content, i.e., it generates a summary leveraging the selected segments [184]. A sketch of the proposed framework's architecture is illustrated in Figure 6.4.

6.3.1 Multimodal sentence selection

The first step of our strategy is to select the most relevant podcast segments for the generation of the summary. This step is performed by a multimodal sentence selection model (steps 1A, 1B and 2 in Figure 6.4), which is trained with the aim of providing a set of sentences that are representative of the podcast content.

Feature extraction The proposed sentence selection model is based on the combination of acoustic and textual features. The text encoder generates sentence representations by using a MPNet [169] model optimized for the generation of semantic-aware text embeddings. Each sentence vector consists of 768 features, and the model can handle sentences up to 512 tokens. The audio features are extracted by leveraging the OpenSmile toolkit, which extracts 88 low-level descriptors from the raw waveform of the audio segments. Features are collected over one second of audio signal (i.e., less than one sentence) and then aggregated over time using the mean, maximum, minimum, and standard deviation functions. The resulting vectors are concatenated to obtain a 352-dimensional feature vector. The multimodal representation of each sentence is obtained by concatenating the audio and text embeddings, i.e., each sentence is represented as a 1120-dimensional vector.

Sentence selection model The selection step relies on a fully-connected architecture that is optimized to score the relevance of each audio-text pair. It is trained, using the regression objective, to predict the sentence-level relevance score, which is a real value in the range $[0,1]$. Similar to MATeR (see Section 6.2), the relevance score for each sentence is computed by using the cosine similarity between the text representation of the sentence and the embedding of the podcast description. The depth of the network is set to seven and the width for the first three layers is set to 1120, while the last four layers have a width of 768. The final layer outputs a real value that corresponds to the relevance score of the audio-text pair.

6.3.2 Podcast summary generation

The output of the sentence selection step is a ranked list of audio-text pairs. Summary generation entails (i) the selection of top-ranked audio snippets to generate an audio summary, and (ii) the selection of top-ranked sentences to generate an abstractive summary of the podcast episode.

Audio summary We take the top-ranked audio segments from the list of audio-text pairs to generate the audio summary. The task organizers set the maximum length of the audio summary to be 60 seconds. Starting from the top, we include the audio segments until the duration of the summary reaches 60 seconds. We re-rank the top-ranked audio fragments based on their original order in the podcast episode. The final summary is then a sequence of relevant audio segments, following the order of their appearance in the podcast episode.

Abstractive summary generation The text summaries are generated using a filtered set of sentences from the podcast episode. Similar to the audio summary, we take the top-ranked sentences from each of the ranked audio-text pairs to obtain a set of sentences that have the highest relevance within the podcast episode.

To generate a fluent and concise summary, we apply an abstractive summarization step on top of the selected sentences. The abstractive summarization system relies on LongFormer Encoder-Decoder architecture (LED) [9] that can handle long text sequences. This model is based on the transformer architecture while introducing some modifications to reduce the computational complexity of the attention mechanism. Instead of computing the attention between each token and all the others in the sequence, the attention is only computed between the current token and a local window of nearby tokens. This allows us to process long sequences of text while significantly reducing the computational cost of both model training and inference.

In order to generate text summaries that closely match the description provided by the podcast author, we fine-tune the pre-trained LED model on the Spotify Podcast dataset [28] which is used for the TREC podcast track. We use the output of the selection step as the training data to fine-tune the model. The maximum number of sentences that can be input into the model is set to 50. During testing, we use the fine-tuned model to generate podcast summaries leveraging the top-ranked sentences derived from the selection process. In the experimental section, we investigate the impact of three parameters for the abstractive summarization model: the minimum and maximum summary length and the number of input sentences.

6.3.3 Experimental evaluation

In order to determine the effectiveness of the proposed systems, track's organizers conduct a manual evaluation of each run submitted by task's participants (i.e., each team can submit up to 4 runs). Human assessors evaluate the overall quality of the written summaries on a four-step scale: *Excellent* (4), *Good* (3), *Fair* (2) and *Bad* (1). They also provide explicit evaluation of some specific characteristics of the summary according to a set of boolean attributes [75].

Q1 Does the summary include names of the main people (hosts, guests, characters) involved or mentioned in the podcast?

Q2 Does the summary give any additional information about the people mentioned (such as their job titles, biographies, personal background, etc)?

Q3 Does the summary include the main topic(s) of the podcast?

Q4 Does the summary tell you anything about the format of the podcast; e.g. whether it's an interview, whether it's a chat between friends, a monologue, etc?

Q5 Does the summary give you more context on the title of the podcast?

Q6 Does the summary contain redundant information? In this specific case, lower scores are better.

Q7 Is the summary written in good English?

Q8 Are the start and end of the summary good sentence and paragraph start and end points?

The audio summary is evaluated by using a binary scale; the assessor provide a positive judgement when the audio captures the essence of the podcast, a negative score otherwise. The questions can be grouped according to their target. The first group (*Q1-Q2*) aims at measuring the effectiveness of the summarization model in capturing the main people involved in the podcast, as well as their attributes. The second group (*Q3-Q5*) evaluates the ability of the proposed system to identify the main topic(s) of the podcast, and to provide relevant context. Finally, *Q6* evaluates the redundancy of the summary while *Q7-Q8* assess its grammar and overall fluency in written English.

Podcast data collection The training data for the podcast summarization task is based on the Spotify Podcast Dataset [28] and contains the audio file and transcript of 100,000 podcast episodes, together with the corresponding descriptions provided by the authors. Following the same approach used in MATeR (see Section 6.2.1), we filter podcast's descriptions to remove advertising and non-relevant information.

The test set, instead, is provided from task organizers and do not overlap with training data. It includes 1,000 podcast episodes, and only the audio files and their transcripts are provided to the participants.

System configuration Participants are allowed to submit up to four runs, each with a different system configuration. The proposed method combines the selection of N top-scored audio-text pairs with the generation of summaries having minimum and maximum length m_l and M_l respectively. Therefore, we submit four systems with different values for N, m_l and M_l to evaluate the contribution of the selection and rewriting phases.

- $N = 25$, $m_l = 32$, $M_L = 128$ (hereafter, $25, 32 - 128$): the input of the summary generation module consists of the 25 top-scored sentences. The final summary has a minimum length of 32 words and can be up to 128 words long.
- $N = 50$, $m_l = 32$, $M_L = 128$ (hereafter, $50, 32 - 128$): same settings as the first system but with the number of selected sentences increased to 50. The selection model in this case provides more information as input to the summary generation module.
- $N = 50$, $m_l = 64$, $M_L = 128$ (hereafter, $50, 64 - 128$): in this case, the length of the output summary is constrained to be at least 64 words, thus forcing the summary to be longer and more descriptive.
- $N = 100$, $m_l = 32$, $M_L = 128$ (hereafter, $100, 32 - 128$): in this run, the number of top-scored sentences is increased to 100 to provide more information to the summary generation module. The selection phase, in this case, is less effective and may provide some redundant or non-relevant information to the summary generation module.

The audio summaries generated by the proposed system do not differ across different runs. Indeed, the summary is automatically generated based on the ranked list of audio-text pairs to met the audio length constraint. Only the textual summary differs between the proposed runs.

Results The assessors were asked to evaluate a subset of 193 podcast episodes randomly sampled from the test set. The results of the proposed system are shown in Table 6.4. Our model is compared against the baseline provided by the task's organizers that extract the first minute of a podcast episode as an audio summary and use its transcription to produce the textual summary.

Analyzing the score for the audio summary, all our submissions outperforms the baseline proposed by the organizers. This could be mainly due to the multimodal

Table 6.4 Human evaluation results. The highest score for each assessment question is highlighted in bold.

Model $(N, m_l - M_L)$	Quality	Q1	Q2	Q3	Q4	Q5	Q6 $\downarrow$	Q7	Q8	Audio
baseline	0.772	**0.549**	**0.326**	0.606	0.427	0.536	0.451	0.456	0.187	0.957
$25, 32-128$	0.974	0.354	0.255	0.645	**0.513**	0.523	0.183	0.806	0.594	0.978
$50, 32-128$	0.860	0.323	0.234	0.615	0.437	0.500	0.204	**0.811**	**0.615**	0.989
$50, 64-128$	**1.010**	0.378	0.285	**0.682**	0.503	**0.562**	0.292	0.715	0.536	0.983
$100, 32-128$	0.917	0.333	0.256	0.606	0.446	0.497	**0.182**	0.776	0.601	**0.994**

nature of our system that uses information from both audio and textual data. Speaker expressions, indeed, are often missed by automatic transcription, but can be exploited by our system to generate more informative and engaging summaries.

Considering the quality score provided by the assessors for the textual summary, we can see that our system is able to improve the quality of the summaries when compared to the baseline. The highest quality score, in this case, is reached by the system with higher m_l threshold for the abstractive summary (i.e., the system is required to generate a summary containing at least 64 tokens). This is in line with what we expect, since the system is able to provide longer summaries and may include more information for the end-user.

All the submitted runs, however, are significantly worse than the baseline considering the evaluation for *Q1-Q2*. Modern abstractive summarization models, indeed, are able to provide fluent and concise summaries (e.g., better scores for *Q6-Q8*) but often they are not factually accurate [115]. The simple baseline proposed by the organizers, which is not based on neural networks, is able to provide more factually accurate summaries. This result is expected, since the baseline does not account the text rephrasing but rather extracts the first sentence of the podcast transcript to form the summary. The use of neural networks to generate abstractive summaries usually leads to higher levels of fluency and conciseness, but requires specific countermeasures to preserve the factual accuracy.

Finally, when compared with the baseline, the summaries generated by our approach achieves higher scores for *Q3-Q5* indicating that our methodology is better at identifying the main topic(s) of the podcast and provides relevant context for the listener. Thus, selecting relevant content from the podcast first to remove redundant content, and then generating a summary which focuses on the relevant parts further validates our *select and rewrite* approach. According to the results of the human assessment, when compared with other participants, our system ranked first based on the quality score of textual summaries while it ranked second when considering audio summaries.

6.4 Challenges and opportunities of multimodal audio analysis

In this chapter we presented our contribution to the analysis of spoken content for the task of podcast summarization. The presented methodologies leverage multimodal audio representations to create a system that efficiently summarizes podcast episodes by extracting the most valuable information, and, potentially, generates more concise and fluent summaries by using modern transformer models. The analysis of spoken content poses several challenges that represent interesting research directions for future works.

6.4.1 Multimodal nature of spoken content

Speech is multimodal by nature, and, as such, should be analyzed according to the multiple sources of information that characterize it. Research in NLP has achieved notable results by developing uni-modal approaches that mainly leverage text. Natural language processing and understanding, however, are rapidly evolving towards multi-modal approaches that are able to exploit multiple sources of information that can contribute to the understanding of the spoken content.

In this context, multimodal approaches can leverage complementary modalities such as text and audio to build richer representations of the content and provide a better understanding of the semantics of the underlying concepts. In this way, the application of language processing techniques to the text modality can be supplemented with the use of specific audio processing techniques. This can contribute to the generation of richer and more complete representations of the spoken content.

The generalization of uni-modal concepts to the multi-modal domain, however, is not trivial, and requires the design of specific solutions that are able to effectively bind and align the complementary sources of information to facilitate the understanding of the speech. Aligning multiple modalities is actually challenging due to the difficulty of determining the mapping between multiple sources of information that can be used to match and align the different modalities [87]. While the alignment of text and visual modalities is a well-established and mature research field that has provided effective solutions to the problem [148, 46], matching text and acoustic data remains an emerging area of research that is still in its infancy.

The use of contrastive learning has shown to be an effective way to simplify the process of aligning multiple modalities. The general idea is to learn data representations in self-supervised manner by providing positive and negative examples to a neural model. The use of those approaches to learn latent representations of text and visual data is considered one of the most effective approaches to their alignment [198]. Recent studies have demonstrated that contrastive learning can be effectively used to align text and audio when combined with ad-hoc self-supervised objectives [159].

6.4.2 Towards textless NLP models

As discussed in Section 6.1.2, spoken content does not necessarily have an accompanying written transcript, which can impede the development of data-driven NLP methods. In addition, spoken content is characterized by complex acoustic phenomena, such as disfluencies, hesitations, and mispronunciations, which are not necessarily reflected in the transcript. Therefore, transcript-based NLP methods may not be able to effectively handle spoken content and may be limited to a small number of tasks and domains.

A recent trend in the NLP community is the development of *textless NLP* methodologies, which aim at developing NLP methods that do not require transcribed text. Future development of the proposed methodology for audio summarization can be extended to a textless framework by developing models for audio-only sequence modeling, where the input to the model is raw speech waveforms, without any accompanying text. The use of these models can both eliminate the requirement of transcripts, that may contain errors, and leverage low-level acoustic feature to learn high-level representations.

Radicalizing the concept behind textless models, it is possible to envision that, in the future, large textless models might be used to *generate* audio summaries (rather than just selecting relevant sections of speech). Sequence-to-sequence models, that are currently the state of the art in text-based summarization tasks, can be adapted to generate audio summaries by optimizing them for the analysis of raw speech waveforms, as opposed to transcribed text. Using as input the speech signal, the model can learn to (i) identify important sections of the speech using acoustic features such as prosody, (ii) learn to conceptually abstract complex acoustic phenomena such as hesitations and stutters, and (iii) generate the audio summary using the voice of the original speaker.

Modeling such systems, however, is a challenging task that requires considerable research efforts. First, the model must be capable of effectively modeling raw audio waveforms, which have a high degree of variability and dimensionality. Second, it must be trained using self-supervised learning, as large-scale annotations for audio summaries are not readily available, and therefore it is not possible to train the model using only manual annotations. Third, the model must be capable of modeling the topic of the speech and generating a summary that is coherent and contains content relevant to its topic. Finally, the generation of speech using a specific target voice is a nontrivial task that relies on modeling the original speaker's voice to generate natural and intelligible speech. In light of the limitations of current voice synthesis systems, this task remains challenging and requires specific attention from the scientific community [23].

Ethical considerations The development of textless NLP architectures, including generative audio models, raises a number of ethical considerations. First, the generation of realistic synthetic speech requires the inclusion of physical characteristics of the voice, such as age, gender, and accents. If the model is not properly trained on a diverse enough dataset, it may result in bias, which may be reflected in the generated speech. Second, the generation of synthetic speech may be used for malicious purposes, such as creating fake audio content. The latter raises a number of ethical considerations, including the possibility of generating misleading and false content and the possibility of violating the privacy of the person that is imitated.

The discussion of future research directions needs to include such concerns, even if they are not directly limited to the application of textless NLP in the context of audio summarization. Indeed, the ethical considerations of developing audio synthesis models are broader, and may be extended to the development of other applications that use synthetic speech as input or output.

Chapter 7

Conclusions

The purpose of this dissertation was to examine the use of deep learning in the domain of Natural Language Processing and its application to the summarization task. Although access to large amounts of information is considered a major advantage of the modern world, it also entails a strain on the cognitive processes of individuals The sheer volume of information available on the Internet, for example, has resulted in information overload, a condition in which individuals may be overwhelmed by the amount of information available. The abundance of content has also led to an increase in the need to automatically distill relevant information from multiple sources.

NLP can help reduce information overload by providing methods that are able to automatically process and understand linguistic information. Throughout this dissertation, we present applications of deep learning, a subset of machine learning that is especially well-suited to process unstructured or complex data, including natural language. We specifically focus on the task of summarization whose aim is to provide a concise and coherent overview of a document while retaining its most important information.

The need for effective summarization models is particularly important in academia, in which scholars are often required to assess large amounts of literature to keep up with the latest advances in their field. Besides understanding and reading the papers, they should be able to quickly identify the key points and main contributions of each article. In this context, we leverage the understanding of citation context to automatically identify the purpose of a citation. By exploiting both the citation context and the full-text of the referenced paper we propose a classification model that is able to detect whether reading the target paper is likely to be useful to understand the citing paper. We then turn our attention on the task of extracting highlights from the full-text of scientific publications, that consist in short sentences, in bullet-point format, that provide a quick overview of the publication's results. By leveraging linguistic and structural characteristics of the text, we propose a machine learning model that can automatically identify highlights within scientific publications. In the academic domain, however, identifying the different aspects of a paper is crucial to a comprehensive understanding of the work. Toward this end, we also propose a methodology that automatically identifies discourse facets in a given publication and produce a summary tailored to each of them. Finally, we explore the applicability of unsupervised summarization models for automatically generating slides presentation for a research paper. Researchers often use slides to disseminate their work, and their automatic generation could greatly reduce their workload.

Information overload, however, is a general phenomenon that does not affect only the academic world. In our daily lives, we are constantly overburdened with information, which can often make it difficult to select the information we need to focus on. This is especially true for online news, where there are so many articles published every day that it is impossible to read them all. Therefore, we address the task of automatically summarizing news stories by considering the temporal aspect of the information. We tackle the timeline summarization task whose goal is to provide a summary for a list of news articles pertaining to a specific topic. First, we summarize the events that occurred at each date, and then we combine the information to create a single timeline that includes both the most significant dates and the most significant events that occurred at those dates. Furthermore, we examine whether cross-lingual settings can be used to improve timeline summarization by leveraging information in multilingual news corpora.

Aside from text, spoken content is another source of information that we use on a regular basis. Increasing availability of multimedia content, such as audio and video, has made it possible to access a vast amount of information that was previously unavailable. Automatically processing this type of content is often more difficult than processing text, as it requires the ability to recognize and understand spoken language. In our contribution, we focus on the problem of automatic summarization of audio content, specifically podcasts. Audio podcasts have become increasingly popular, as they provide a convenient way to consume information while commuting or doing other activities. Due to the wide variety of podcasts available, it is difficult for listeners to retrieve the most relevant content. Therefore, automatic summarization of podcasts helps listeners quickly find the information they are looking for. We tackle the problem using both extractive and abstractive summarization approaches. In the extractive approach, we leverage a multimodal ranking model that exploits both the audio signal and the speech transcription to identify the most important segments. The abstractive approach, first selects important segments using multimodal ranking, and then leverages them to generate a summary using a sequence-to-sequence model.

In conclusion, we presented several methods for automatic summarization of text and audio content that can be used to effectively handle information overload. We found that deep learning techniques efficiently and effectively learn representations of text and audio suitable for the summarization task. The versatility of deep learning for natural language understanding, and specifically its potential for automatic summarization, was demonstrated by investigating a variety of different domains. These methods can help us cope better with information overload and make better use of the overwhelming amount of information available to us.

7.1 Future research directions

In addition to the future research directions that are already outlined for each chapter, we would like to mention a few additional general areas where we believe further research should be conducted.

The computational cost of training and testing the models is a common challenge in all the tasks we have covered and, more broadly, in all methodologies that use deep learning models. As the models become more complex, training becomes more time-consuming, and, in some cases, it becomes prohibitive to retrain the models from scratch. *Transfer learning* [38, 3] could provide a solution, which consists of reusing models that have already been trained on large amounts of data. The approach is very effective and has largely been responsible for deep learning's success in the past few years, but it also has some limitations. Firstly, even fine-tuning a large, state-of-the-art model requires considerable computational resources, which may not be available to many researchers. Secondly, fine-tuning a pre-trained model might not be the ideal solution for all tasks, as, in some cases, using different pre-training objectives or starting from scratch may be more effective [57]. In this context, it is important to develop methods that can train large models efficiently or can reduce the model size without significantly affecting performance [163].

Another relevant direction for future work is to develop methods that are able to generalize to different modalities. In the context of podcast summarization, we have already explored the use of audio and text modalities. However, other tasks might also benefit from using multiple modalities, such as visual and textual modalities for video summarization. Considering deep learning research at a high level, most of the architectures still rely on induction biases to achieve good generalization on specific modalities. In the future, it would be possible to design modality-agnostic machine learning models which can be generalized to any type of input data by developing techniques that are not biased towards a specific modality. Perceiver [71] and Data2Vec [2] are recent examples of architectures that can learn data representations that are agnostic to the input data modality.

References

[1] Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., and Sakai, T. (2015). Trec 2014 temporal summarization track overview. Technical report, NATIONAL INST OF STANDARDS AND TECHNOLOGY GAITHERSBURG MD.

[2] Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., and Auli, M. (2022). Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*.

[3] Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

[4] Bai, Z. and Zhang, X.-L. (2021). Speaker recognition based on deep learning: An overview. *Neural Networks*, 140:65–99.

[5] Barrios, F., López, F., Argerich, L., and Wachenchauzer, R. (2016). Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*.

[6] Battaglino, D., Lepauloux, L., Evans, N., Mougins, F., and Biot, F. (2016). Acoustic scene classification using convolutional neural networks. *IEEE AASP Challenge on Detec*.

[7] Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563.

[8] Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

[9] Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

[10] Berrebbi, D., Huynh, N., and Balalau, O. (2022). GraphCite: Citation Intent Classification in Scientific Publications via Graph Embeddings. In *2nd International Workshop on Scientific Knowledge: Representation, Discovery, and Assessment*, Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), Lyon / Virtual, France.

[11] Binh Tran, G., Alrifai, M., and Quoc Nguyen, D. (2013). Predicting relevant news events for timeline summaries. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 91–92.

[12] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

[13] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

[14] Cachola, I., Lo, K., Cohan, A., and Weld, D. (2020). TLDR: Extreme summarization of scientific documents. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4766–4777, Online. Association for Computational Linguistics.

[15] Cagliero, L., Garza, P., and Baralis, E. (2019). Elsa: A multilingual document summarization algorithm based on frequent itemsets and latent semantic analysis. *ACM Trans. Inf. Syst.*, 37(2).

[16] Cagliero, L. and La Quatra, M. (2020). Extracting highlights of scientific articles: A supervised summarization approach. *Expert Systems with Applications*, 160:113659.

[17] Cagliero, L. and La Quatra, M. (2021a). Automatic slides generation in the absence of training data. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 103–108. IEEE.

[18] Cagliero, L. and La Quatra, M. (2021b). Inferring multilingual domain-specific word embeddings from large document corpora. *IEEE Access*, 9:137309–137321.

[19] Campos, V., Campos, R., Mota, P., and Jorge, A. (2022). Tweet2story: A web app to extract narratives from twitter. In *European Conference on Information Retrieval*, pages 270–275. Springer.

[20] Castells, M. (1997). An introduction to the information age. *City*, 2(7):6–16.

[21] Chandna, P., Miron, M., Janer, J., and Gómez, E. (2017). Monoaural audio source separation using deep convolutional neural networks. In *International conference on latent variable analysis and signal separation*, pages 258–266. Springer.

[22] Chandrasekaran, M. K., Yasunaga, M., Radev, D. R., Freitag, D., and Kan, M. (2019). Overview and results: Cl-scisumm shared task 2019. In Chandrasekaran, M. K. and Mayr, P., editors, *Proceedings of the 4th Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2019) co-located with the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019), Paris, France, July 25, 2019*, volume 2414 of *CEUR Workshop Proceedings*, pages 153–166. CEUR-WS.org.

[23] Chen, B., Du, C., and Yu, K. (2022). Neural fusion for voice cloning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pages 1–1.

[24] Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., et al. (2021). Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *arXiv preprint arXiv:2110.13900*.

[25] Chen, X., Chan, Z., Gao, S., Yu, M.-H., Zhao, D., and Yan, R. (2019). Learning towards abstractive timeline summarization. In *IJCAI*, pages 4939–4945.

[26] Chieu, H. L. and Lee, Y. K. (2004). Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–432.

[27] Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

[28] Clifton, A., Reddy, S., Yu, Y., Pappu, A., Rezapour, R., Bonab, H., Eskevich, M., Jones, G., Karlgren, J., Carterette, B., and Jones, R. (2020). 100,000 podcasts: A spoken English document corpus. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5903–5917, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[29] Cohan, A., Ammar, W., van Zuylen, M., and Cady, F. (2019). Structural scaffolds for citation intent classification in scientific publications. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3586–3596, Minneapolis, Minnesota. Association for Computational Linguistics.

[30] Cohan, A., Dernoncourt, F., Kim, D. S., Bui, T., Kim, S., Chang, W., and Goharian, N. (2018). A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

[31] Cohan, A. and Goharian, N. (2015). Scientific article summarization using citation-context and article's discourse structure. In *Proceedings of the 2015*

Conference on Empirical Methods in Natural Language Processing, pages 390–400, Lisbon, Portugal. Association for Computational Linguistics.

[32] Cohan, A. and Goharian, N. (2018). Scientific document summarization via citation contextualization and scientific discourse. *International Journal on Digital Libraries*, 19(2):287–303.

[33] Collins, E., Augenstein, I., and Riedel, S. (2017). A supervised approach to extractive summarisation of scientific papers. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 195–205, Vancouver, Canada. Association for Computational Linguistics.

[34] Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

[35] Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical part-of-speech tagger. In *Third conference on applied natural language processing*, pages 133–140.

[36] Dale, R. (2021). Gpt-3: What's it good for? *Natural Language Engineering*, 27(1):113–118.

[37] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

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

[39] Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. (2014). Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1370–1380, Baltimore, Maryland. Association for Computational Linguistics.

[40] Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

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

[42] Eyben, F., Wöllmer, M., and Schuller, B. (2010). Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462.

[43] Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2021). Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

[44] Ferrod, R., Caro, L. D., and Schifanella, C. (2021). Structured semantic modeling of scientific citation intents. In *European Semantic Web Conference*, pages 461–476. Springer.

[45] Ferrod, R., Schifanella, C., Caro, L. D., and Cataldi, M. (2019). Disclosing citation meanings for augmented research retrieval and exploration. In *European Semantic Web Conference*, pages 101–115. Springer.

[46] Frank, S., Bugliarello, E., and Elliott, D. (2021). Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[47] Fu, Y., Zhou, H., Chen, J., and Li, L. (2019). Rethinking text attribute transfer: A lexical analysis. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 24–33, Tokyo, Japan. Association for Computational Linguistics.

[48] Fung, P., Ngai, G., and Cheung, C.-S. (2003). Combining optimal clustering and hidden Markov models for extractive summarization. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, pages 21–28, Sapporo, Japan. Association for Computational Linguistics.

[49] Gao, Y., Zhao, W., and Eger, S. (2020). SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

[50] Gholipour Ghalandari, D. (2017). Revisiting the centroid-based method: A strong baseline for multi-document summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 85–90, Copenhagen, Denmark. Association for Computational Linguistics.

[51] Gholipour Ghalandari, D. and Ifrim, G. (2020). Examining the state-of-the-art in news timeline summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1322–1334, Online. Association for Computational Linguistics.

[52] Gokhan, T., Smith, P., and Lee, M. (2021). Extractive financial narrative summarisation using SentenceBERT based clustering. In *Proceedings of the*

3rd Financial Narrative Processing Workshop, pages 94–98, Lancaster, United Kingdom. Association for Computational Linguistics.

[53] Golub, G. H. and Reinsch, C. (1971). Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer.

[54] Gong, Y., Chung, Y.-A., and Glass, J. (2021a). AST: Audio Spectrogram Transformer. In *Proc. Interspeech 2021*, pages 571–575.

[55] Gong, Y., Lai, C.-I. J., Chung, Y.-A., and Glass, J. (2021b). Ssast: Self-supervised audio spectrogram transformer. *arXiv preprint arXiv:2110.09784*.

[56] Gupta, V., Bharti, P., Nokhiz, P., and Karnick, H. (2021). SumPubMed: Summarization dataset of PubMed scientific articles. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 292–303, Online. Association for Computational Linguistics.

[57] Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

[58] Han, J. M., Babuschkin, I., Edwards, H., Neelakantan, A., Xu, T., Polu, S., Ray, A., Shyam, P., Ramesh, A., Radford, A., et al. (2021). Unsupervised neural machine translation with generative language models only. *arXiv preprint arXiv:2110.05448*.

[59] Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

[60] Hemp, P. (2009). Death by information overload. *Harvard business review*, 87(9):82–9.

[61] Hernández-Alvarez, M. and Gómez, J. M. (2015). Citation impact categorization: for scientific literature. In *2015 IEEE 18th International Conference on Computational Science and Engineering*, pages 307–313. IEEE.

[62] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[63] Hofmann, K., Marakasova, A., Baumann, A., Neidhardt, J., and Wissik, T. (2020). Comparing lexical usage in political discourse across diachronic corpora. In *Proceedings of the Second ParlaCLARIN Workshop*, pages 58–65, Marseille, France. European Language Resources Association.

[64] Hou, Y., Jochim, C., Gleize, M., Bonin, F., and Ganguly, D. (2021). TDMSci: A specialized corpus for scientific literature entity tagging of tasks datasets and metrics. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 707–714, Online. Association for Computational Linguistics.

[65] Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. (2021). Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

[66] Hsu, W.-T., Lin, C.-K., Lee, M.-Y., Min, K., Tang, J., and Sun, M. (2018). A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141.

[67] Hu, Y. and Wan, X. (2014). Ppsgen: Learning-based presentation slides generation for academic papers. *IEEE transactions on knowledge and data engineering*, 27(4):1085–1097.

[68] Huang, R., Zhao, Z., Liu, J., Liu, H., Ren, Y., Zhang, L., and He, J. (2022). Transpeech: Speech-to-speech translation with bilateral perturbation. *arXiv preprint arXiv:2205.12523*.

[69] Ihsan, I. and Qadir, M. A. (2021). An nlp-based citation reason analysis using ccro. *Scientometrics*, 126(6):4769–4791.

[70] Iqbal, S., Hassan, S.-U., Aljohani, N. R., Alelyani, S., Nawaz, R., and Bornmann, L. (2021). A decade of in-text citation analysis based on natural language processing and machine learning techniques: An overview of empirical studies. *Scientometrics*, 126(8):6551–6599.

[71] Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., and Carreira, J. (2021). Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR.

[72] Jebari, C., Herrera-Viedma, E., and Cobo, M. J. (2021). The use of citation context to detect the evolution of research topics: a large-scale analysis. *Scientometrics*, 126(4):2971–2989.

[73] Jurgens, D., Kumar, S., Hoover, R., McFarland, D., and Jurafsky, D. (2018). Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406.

[74] Karim, M., Missen, M. M. S., Umer, M., Sadiq, S., Mohamed, A., and Ashraf, I. (2022). Citation context analysis using combined feature embedding and deep convolutional neural network model. *Applied Sciences*, 12(6):3203.

[75] Karlgren, J., Jones, R., Carterette, B., Clifton, A., Eskevich, M., Gareth J. F., J., Reddy, S., Tanaka, E., and Tanveer, M. I. (2021). Trec 2021 podcasts track overview.

[76] Kessler, R., Tannier, X., Hagège, C., Moriceau, V., and Bittar, A. (2012). Finding salient dates for building thematic timelines. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 730–739, Jeju Island, Korea. Association for Computational Linguistics.

[77] Kim, B., Kim, H., and Kim, G. (2019). Abstractive summarization of Reddit posts with multi-level memory networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2519–2531, Minneapolis, Minnesota. Association for Computational Linguistics.

[78] Kitaev, N., Kaiser, L., and Levskaya, A. (2019). Reformer: The efficient transformer. In *International Conference on Learning Representations*.

[79] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

[80] Krapivin, M., Autayeu, A., Marchese, M., Blanzieri, E., and Segata, N. (2010). Keyphrases extraction from scientific documents: improving machine learning approaches with natural language processing. In *International Conference on Asian Digital Libraries*, pages 102–111. Springer.

[81] Kreuk, F., Polyak, A., Copet, J., Kharitonov, E., Nguyen, T.-A., Rivière, M., Hsu, W.-N., Mohamed, A., Dupoux, E., and Adi, Y. (2021). Textless speech emotion conversion using decomposed and discrete representations. *arXiv preprint arXiv:2111.07402*.

[82] Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR.

[83] La Quatra, M., Cagliero, L., and Baralis, E. (2019). Poli2sum@ cl-scisumm-19: Identify, classify, and summarize cited text spans by means of ensembles of supervised models. In *BIRNDL@ SIGIR*.

[84] La Quatra, M., Cagliero, L., and Baralis, E. (2020). Exploiting pivot words to classify and summarize discourse facets of scientific papers. *Scientometrics*, 125(3):3139–3157.

[85] La Quatra, M., Cagliero, L., and Baralis, E. (2021a). Leveraging full-text article exploration for citation analysis. *Scientometrics*, 126(10):8275–8293.

[86] La Quatra, M., Cagliero, L., Baralis, E., Messina, A., and Montagnuolo, M. (2021b). Summarize dates first: A paradigm shift in timeline summarization. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 418–427.

[87] La Quatra, M., Vaiani, L., Cagliero, L., and Garza, P. (2022). Bi-modal architectures for deeper user preference understanding from spoken content.

[88] Lakhotia, K., Kharitonov, E., Hsu, W.-N., Adi, Y., Polyak, A., Bolte, B., Nguyen, T.-A., Copet, J., Baevski, A., Mohamed, A., and Dupoux, E. (2021). On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354.

[89] Lamsiyah, S., El Mahdaouy, A., Espinasse, B., and Ouatik, S. E. A. (2021). An unsupervised method for extractive multi-document summarization based on centroid approach and sentence embeddings. *Expert Systems with Applications*, 167:114152.

[90] Landauer, T. K., Foltz, P. W., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284.

[91] Larson, M. and Jones, G. J. (2012). Spoken content retrieval: A survey of techniques and technologies. *Foundations and Trends in Information Retrieval*, 5(4—5):235–422.

[92] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

[93] Li, D., Chen, T., Tung, A., and Chilton, L. B. (2021a). Hierarchical summarization for longform spoken dialog. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 582–597.

[94] Li, L., Zhu, Y., Xie, Y., Huang, Z., Liu, W., Li, X., and Liu, Y. (2019). Cist@ clscisumm-19: Automatic scientific paper summarization with citances and facets. *BIRNDL@ SIGIR*, 54.

[95] Li, M., Ma, T., Yu, M., Wu, L., Gao, T., Ji, H., and McKeown, K. (2021b). Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6443–6456, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[96] Liao, Y., Wang, S., and Lee, D. (2021). Wilson: A divide and conquer approach for fast and effective news timeline summarization. In *EDBT*, pages 635–645.

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

[98] Lin, H. and Bilmes, J. (2010). Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920, Los Angeles, California. Association for Computational Linguistics.

[99] Lin, H. and Bilmes, J. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 510–520, Portland, Oregon, USA. Association for Computational Linguistics.

[100] Lin, H. and Ng, V. (2019). Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9815–9822.

[101] Lin, J., Roegiest, A., Tan, L., McCreadie, R., Voorhees, E. M., and Diaz, F. (2016). Overview of the trec 2016 real-time summarization track. In *TREC*.

[102] Liu, T.-E., Liu, S.-H., and Chen, B. (2019a). A hierarchical neural summarization framework for spoken documents. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7185–7189.

[103] Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

[104] Liu, Y. and Lapata, M. (2019). Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

[105] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[106] Loshchilov, I. and Hutter, F. (2018). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

[107] Manakul, P., Gales, M. J., and Wang, L. (2020). Abstractive spoken document summarization using hierarchical model with multi-stage attention diversity optimization. *Proc. Interspeech 2020*, pages 4248–4252.

[108] Manning, C. and Schutze, H. (1999). *Foundations of statistical natural language processing*. MIT press.

[109] Mao, Q., Li, J., Wang, J., Li, X., Hao, P., Wang, L., and Wang, Z. (2022a). Explicitly modeling importance and coherence for timeline summarization. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8062–8066. IEEE.

[110] Mao, Y., Zhong, M., and Han, J. (2022b). Citesum: Citation text-guided scientific extreme summarization and low-resource domain adaptation. *arXiv preprint arXiv:2205.06207*.

[111] Martschat, S. and Markert, K. (2017). Improving ROUGE for timeline summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 285–290, Valencia, Spain. Association for Computational Linguistics.

[112] Martschat, S. and Markert, K. (2018). A temporally sensitive submodularity framework for timeline summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 230–240, Brussels, Belgium. Association for Computational Linguistics.

[113] Masataki, H. and Sgisaka, Y. (1996). Variable-order n-gram generation by word-class splitting and consecutive word grouping. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 188–191. IEEE.

[114] Mashechkin, I., Petrovskiy, M., Popov, D., and Tsarev, D. V. (2011). Automatic text summarization using latent semantic analysis. *Programming and Computer Software*, 37(6):299–305.

[115] Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. (2020). On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

[116] McCarthy, J. F. and Lehnert, W. G. (1995). Using decision trees for conference resolution. In *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 2*, pages 1050–1055.

[117] McCreadie, R., Buntain, C., and Soboroff, I. (2019). Trec incident streams: Finding actionable information on social media.

[118] McCreadie, R., Macdonald, C., and Ounis, I. (2014). Incremental update summarization: Adaptive sentence selection based on prevalence and novelty. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pages 301–310.

[119] Mei, Q. and Zhai, C. (2008). Generating impact-based summaries for scientific literature. In *Proceedings of ACL-08: HLT*, pages 816–824, Columbus, Ohio. Association for Computational Linguistics.

[120] Meireles, M. R. G. and Cendón, B. V. (2017). Citation-based document categorization: an approach using artificial neural networks. *Qualitative and Quantitative Methods in Libraries*, pages 71–79.

[121] Mercer, R. E. and Di Marco, C. (2004). A design methodology for a biomedical literature indexing tool using the rhetoric of science. In *HLT-NAACL 2004 Workshop: Linking Biological Literature, Ontologies and Databases*, pages 77–84, Boston, Massachusetts, USA. Association for Computational Linguistics.

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

[123] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In Bengio, Y. and LeCun, Y., editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[124] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari.

[125] Millar, E., Shen, D., Liu, J., and Nicholas, C. (2000). Performance and scalability of a large-scale n-gram based information retrieval system. *Journal of digital information*, 1(5).

[126] Mohan, M. J., Sunitha, C., Ganesh, A., and Jaya, A. (2016). A study on ontology based abstractive summarization. *Procedia Computer Science*, 87:32–37.

[127] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

[128] Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. (2021). Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

[129] Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-first AAAI conference on artificial intelligence*.

[130] Narayan, S., Cohen, S. B., and Lapata, M. (2018). Ranking sentences for extractive summarization with reinforcement learning. In *NAACL-HLT*.

[131] Nenkova, A. and McKeown, K. (2012). A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer.

[132] Nguyen, K. and Daumé III, H. (2019). Global Voices: Crossing borders in automatic news summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 90–97, Hong Kong, China. Association for Computational Linguistics.

[133] Nguyen, K.-H., Tannier, X., and Moriceau, V. (2014). Ranking multidocument event descriptions for building thematic timelines. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1208–1217, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

[134] Nguyen, T. A., Kharitonov, E., Copet, J., Adi, Y., Hsu, W.-N., Elkahky, A., Tomasello, P., Algayres, R., Sagot, B., Mohamed, A., et al. (2022). Generative spoken dialogue language modeling. *arXiv preprint arXiv:2203.16502*.

[135] Nicholson, J. M., Mordaunt, M., Lopez, P., Uppala, A., Rosati, D., Rodrigues, N. P., Grabitz, P., and Rife, S. C. (2021). Scite: A smart citation index that displays the context of citations and classifies their intent using deep learning. *Quantitative Science Studies*, 2(3):882–898.

[136] Noreen, E. W. (1989). *Computer-intensive methods for testing hypotheses*. Wiley New York.

[137] Oya, T., Mehdad, Y., Carenini, G., and Ng, R. (2014). A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

[138] Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia.

[139] Pagliardini, M., Gupta, P., and Jaggi, M. (2018). Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

[140] Pasquali, A., Campos, R., Ribeiro, A., Santana, B., Jorge, A., and Jatowt, A. (2021). Tls-covid19: A new annotated corpus for timeline summarization. In *European Conference on Information Retrieval*, pages 497–512. Springer.

[141] Paulus, R., Xiong, C., and Socher, R. (2018). A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

[142] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[143] Polyak, A., Adi, Y., Copet, J., Kharitonov, E., Lakhotia, K., Hsu, W.-N., Mohamed, A., and Dupoux, E. (2021). Speech resynthesis from discrete disentangled self-supervised representations. In *INTERSPEECH 2021-Annual Conference of the International Speech Communication Association*.

[144] Pride, D. and Knoth, P. (2020). An authoritative approach to citation classification. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, pages 337–340.

[145] Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R., and Zhou, M. (2020). ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

[146] Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

[147] Radev, D. R., Jing, H., Styś, M., and Tam, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

[148] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

[149] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training.

[150] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

[151] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

[152] Ramirez-Orta, J. and Milios, E. (2021). Unsupervised document summarization using pre-trained sentence embeddings and graph centrality. In *Proceedings of the Second Workshop on Scholarly Document Processing*, pages 110–115.

[153] Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

[154] Reimers, N. and Gurevych, I. (2020). Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

[155] Rethmeier, N. and Augenstein, I. (2021). A primer on contrastive pretraining in language processing: Methods, lessons learned and perspectives. *arXiv preprint arXiv:2102.12982*.

[156] Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1995). Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

[157] Roman, M., Shahid, A., Khan, S., Koubaa, A., and Yu, L. (2021). Citation intent classification using word embedding. *IEEE Access*, 9:9982–9995.

[158] Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

[159] Sachidananda, V., Tseng, S.-Y., Marchi, E., Kajarekar, S., and Georgiou, P. (2022). Calm: Contrastive aligned audio-language multirate and multimodal representations. *arXiv preprint arXiv:2202.03587*.

[160] Sai, A. B., Mohankumar, A. K., and Khapra, M. M. (2022). A survey of evaluation metrics used for nlg systems. *ACM Comput. Surv.*, 55(2).

[161] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

[162] Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

[163] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

[164] Sawhney, R., Wadhwa, A., Agarwal, S., and Shah, R. R. (2021). FAST: Financial news and tweet based time aware network for stock trading. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2164–2175, Online. Association for Computational Linguistics.

[165] Sefid, A. and Wu, J. (2019). Automatic slide generation for scientific papers. In *Third International Workshop on Capturing Scientific Knowledge co-located with the 10th International Conference on Knowledge Capture (K-CAP 2019), SciKnow@ K-CAP 2019*.

[166] Sharma, R., Palaskar, S., Black, A. W., and Metze, F. (2022). End-to-end speech summarization using restricted self-attention. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8072–8076.

[167] Siddharthan, A. and Teufel, S. (2007). Whose idea was this, and why does it matter? attributing scientific work to citations. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 316–323, Rochester, New York. Association for Computational Linguistics.

[168] Sollaci, L. B. and Pereira, M. G. (2004). The introduction, methods, results, and discussion (imrad) structure: a fifty-year survey. *Journal of the medical library association*, 92(3):364.

[169] Song, K., Tan, X., Qin, T., Lu, J., and Liu, T.-Y. (2020). Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

[170] Steen, J. and Markert, K. (2019). Abstractive timeline summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 21–31, Hong Kong, China. Association for Computational Linguistics.

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

[172] Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. (2020). Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

[173] Strötgen, J. and Gertz, M. (2010). HeidelTime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324, Uppsala, Sweden. Association for Computational Linguistics.

[174] Suen, C. Y. (1979). N-gram statistics for natural language understanding and text processing. *IEEE transactions on pattern analysis and machine intelligence*, (2):164–172.

[175] Swan, R. and Allan, J. (2000). Automatic generation of overview timelines. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56.

[176] Tan, J., Wan, X., and Xiao, J. (2017). Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

[177] Teufel, S., Siddharthan, A., and Tidhar, D. (2006). Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110, Sydney, Australia. Association for Computational Linguistics.

[178] Tixier, A., Meladianos, P., and Vazirgiannis, M. (2017). Combining graph degeneracy and submodularity for unsupervised extractive summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 48–58, Copenhagen, Denmark. Association for Computational Linguistics.

[179] Toffler, A. (1970). Future shock, 1970. *Sydney. Pan.*

[180] Tran, G., Alrifai, M., and Herder, E. (2015a). Timeline summarization from relevant headlines. In *European Conference on Information Retrieval*, pages 245–256. Springer.

[181] Tran, G., Herder, E., and Markert, K. (2015b). Joint graphical models for date selection in timeline summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1598–1607, Beijing, China. Association for Computational Linguistics.

[182] Tran, G. B., Tran, T. A., Tran, N.-K., Alrifai, M., and Kanhabua, N. (2013). Leveraging learning to rank in an optimization framework for timeline summarization. In *SIGIR 2013 Workshop on Time-aware Information Access (TAIA*.

[183] TURING, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, LIX(236):433–460.

[184] Vaiani, L., La Quatra, M., Cagliero, L., and Garza, P. (2021). Polito at trec 2021 podcast summarization track.

[185] Vaiani, L., La Quatra, M., Cagliero, L., and Garza, P. (2022). Leveraging multimodal content for podcast summarization. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, pages 863–870.

[186] Vartakavi, A. and Garg, A. (2020). Podsumm–podcast audio summarization. *arXiv preprint arXiv:2009.10315*.

[187] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[188] Wang, H.-C., Chen, C.-C., and Li, T.-W. (2022a). Automatic content curation of news events. *Multimedia Tools and Applications*, 81(8):10445–10467.

[189] Wang, K., Quan, X., and Wang, R. (2019a). BiSET: Bi-directional selective encoding with template for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2153–2162, Florence, Italy. Association for Computational Linguistics.

[190] Wang, L., Luc, P., Wu, Y., Recasens, A., Smaira, L., Brock, A., Jaegle, A., Alayrac, J.-B., Dieleman, S., Carreira, J., et al. (2022b). Towards learning universal audio representations. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4593–4597. IEEE.

[191] Wang, M., Leng, D., Ren, J., Zeng, Y., and Chen, G. (2019b). Sentiment classification based on linguistic patterns in citation context. *Current Science (00113891)*, 117(4).

[192] Weaver, W. (1952). Translation. In *Proceedings of the Conference on Mechanical Translation*.

[193] Wyse, L. (2017). Audio spectrogram representations for processing with convolutional neural networks. In *Proceedings of the First International Conference on Deep Learning and Music*, pages 37–41.

[194] Xiao, W., Beltagy, I., Carenini, G., and Cohan, A. (2021). Primer: Pyramid-based masked sentence pre-training for multi-document summarization. *arXiv preprint arXiv:2110.08499*.

[195] Xiao, Y. and Wang, W. Y. (2021). On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744, Online. Association for Computational Linguistics.

[196] Yang, Z., Zhu, C., Gmyr, R., Zeng, M., Huang, X., and Darve, E. (2020). Ted: A pretrained unsupervised summarization model with theme modeling and denoising. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1865–1874.

[197] Yasunaga, M., Kasai, J., Zhang, R., Fabbri, A. R., Li, I., Friedman, D., and Radev, D. R. (2019). Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7386–7393.

[198] Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. (2022). Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*.

[199] Yu, Y., Jatowt, A., Doucet, A., Sugiyama, K., and Yoshikawa, M. (2021). Multi-TimeLine summarization (MTLS): Improving timeline summarization by generating multiple summaries. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 377–387, Online. Association for Computational Linguistics.

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

[201] Zhang, M., Zhou, Y., Zhao, L., and Li, H. (2021). Transfer learning from speech synthesis to voice conversion with non-parallel training data. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1290–1302.

[202] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019a). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

[203] Zhang, X., Wei, F., and Zhou, M. (2019b). HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

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

[205] Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M., and Zhao, T. (2018). Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663.

[206] Zhu, C., Yang, Z., Gmyr, R., Zeng, M., and Huang, X. (2021). *Leveraging Lead Bias for Zero-Shot Abstractive News Summarization*, page 1462–1471. Association for Computing Machinery, New York, NY, USA.

[207] Zopf, M., Loza Mencía, E., and Fürnkranz, J. (2016). Sequential clustering and contextual importance measures for incremental update summarization. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1071–1082, Osaka, Japan. The COLING 2016 Organizing Committee.