DOCUSAGE: HARNESSING HIERARCHICAL CLUSTERING IN SALIENCE-DRIVEN NARRATIVE SYNTHESIS

A THESIS SUBMITTED TO
THE GRADUATE DIVISION
OF THE
UNIVERSITY OF HAWAIʻI AT MĀNOA
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

By

Akib Sadmanee

JULY 2024

Thesis Committee:

Mahdi Belcaid, Chairperson
Jason Leigh
Peter Washington

Keywords: Natural language processing, Dataset synthesis, Narrative synthesis, Text summarization

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor, Dr. Mahdi Belcaid, for his invaluable guidance and support throughout this research. He has been a true mentor guiding me through the maze of research while teaching me how to navigate myself.

I am also profoundly thankful to my committee members, Dr. Jason Leigh and Dr. Peter Washington, for their thought-provoking questions, constructive feedback, and crucial suggestions.

Special thanks to my labmates at the Scalable Analytics and Informatics Lab (SAIL) for their camaraderie, support, and stimulating discussions that have enriched my research experience.

My family has played a very important role in keeping my mental fortitude and I am grateful to them for always standing by me in my ups and downs.

Lastly, I thank ChatGPT for making my writing experience much smoother and easier. It has been a great help to rewrite the paragraphs in a more formal and cohesive language.

ABSTRACT

Text summarization remains a crucial yet challenging task in natural language processing, especially as the volume of text data grows exponentially. This thesis introduces Sumsage, a new optimization-based text summarization method that synthesizes concise yet informative summaries. Our work presents several notable contributions to the field. We developed the Syn-D-sum dataset from the CNN/DailyMail dataset, creating a robust resource for training and evaluating summarization models. We also propose the Sumsage algorithm, which leverages hierarchical clustering to extract key sentences and construct coherent summaries, closely emulating human summarizers. Additionally, we designed two new evaluation methods: the Symphony penalty and the Captured Importance Quantification scores, which assess the quality of generated summaries by considering both narrative structure and sentence order. Sumsage's dynamic tree structure and hierarchical clustering approach enable efficient and scalable summarization while maintaining contextual relevance and minimizing hallucination. Additionally, our experiments show that Sumsage yields superior performance over GPT-3.5-turbo, generating summaries similar to those written by humans and capturing more essential information. Sumsage represents a novel advancement in text summarization, offering a robust and interpretable method for generating high-quality summaries. This approach not only addresses current challenges but also lays the foundation for future innovations in narrative synthesis and evaluation.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Text Summarization

In recent years the task of condensing large texts has attracted substantial interest in Natural Language Processing (NLP). This task, also known as text summarization, is becoming increasingly important because of the exponential growth of available text data on the internet.

There are two mainstream approaches to text summarization: extractive and abstractive summarization [83]. Extractive summarization extracts the relevant sentences from the input document and combines them into a summary without changing the sentences at all [26]. To represent it more formally, let's consider a single document D consisting of n sentences: $D = \{s_1^1, s_2^1, ..., s_n^1\}$. The objective of extractive summarization is to compile a concise summary S, where S is a subset of D and the cardinality of S is m ($m \ll n$), by extracting these sentences directly from the source document.

The abstractive summarization process, on the other hand, is the process of generating a summary using the features of the input document and not the exact words or sentences [28]. The generated summary may or may not contain information exactly from the input document. Even though the model is prone to hallucination and missing key concepts, it generates a better summary than the extractive summarizers. The objective of abstractive summarization is to generate a concise summary S consisting of m sentences. Each sentence in the summary is a newly constructed sentence that captures the essential information from D. In contrast to extractive summarization, where $S \subset D$, for abstractive summarization, S is a new set of sentences that may not directly overlap with any specific subset of D.

Despite their effectiveness, both approaches have some inherent limitations, i.e. ab-

stractive summarizers go beyond the provided context or miss important information, and extractive summarizers do not generate cohesive summaries [3]. However, hybrid summarization combines its advantages and mitigates most of its weaknesses to generate summaries with high quality and coherence [99]. Although hybrid summarization, also known as the "extract-then-abstract" model received limited attention before 2022 [9, 76], it began to gradually attract interest thereafter because of the boom in large language models which makes it a very efficient method to extract important sentences from a document and weave them into a coherent passage [103]. As a consequence, narrative extraction and narrative synthesis have currently become popular in the research world as well [116, 82].

1.2 Large Language Models

Language Models have evolved from simple statistical models [20] to complex deep neural networks with over three hundred billion parameters [5]. This evolution has enabled them to generate natural language with high proficiency and accuracy. Large Language Models (LLMs) are designed with multiple layers of transformers designed to handle sequential data and consider the relationships between different parts of the input text. A transformer is a neural network architecture that relies on self-attention mechanisms to assign different weights to various parts of the input data, allowing it to capture intricate dependencies across the sequence effectively. It replaces traditional recurrent layers with attention layers, enabling parallel processing and improving performance on tasks such as language modeling and translation [94]. The transformer layers enable the LLMs to capture long-ranged contextual information. Moreover, during the fine-tuning phase, LLMs can be trained on task-specific datasets to generate contextually appropriate and relevant outputs. LLMs, such as OpenAI's GPT series [11], Google's Gemini [92], and other state-of-the-art models, have revolutionized natural language processing (NLP) by demonstrating the ability to perform a wide range

of tasks with human-like fluency and accuracy. These models are pre-trained on a vast amount of diverse datasets comprising text from books, articles, websites, and other textual sources. This pre-training phase allows LLMs to learn the complexity of natural language, including grammar, semantics, and contextual relationships, thus equipping them with a broad understanding of various topics.

LLMs can be fine-tuned for text summarisation to extract salient sentences and generate coherent summaries that mimic human-written text [39]. This fine-tuning process enhances the models' ability to handle specific nuances and requirements of summarization tasks. LLMs can effectively handle text summarization tasks by generating coherent and concise abstractive summaries [108, 8]. They offer customization options and flexibility in summary generation through fine-tuning, making them powerful tools for various natural language processing tasks. However, the integration of LLMs into summarization systems introduces several challenges, primarily due to their extensive knowledge base and the intricate ways they interpret prompts.

One major issue with LLMs is the phenomenon of hallucination. Because these models are trained on vast and diverse corpora, they inherently possess information about numerous topics. When tasked with summarization, LLMs may incorporate information that is not present in the input document. This is called hallucination in the field of natural language processing [104]. This issue arises because the models' wide-ranging knowledge base can sometimes overshadow the specific content of the document being summarized. Moreover, LLMs operate as black boxes, making it difficult to understand and control the criteria they use to determine the importance of information in a text. This opacity can result in the omission of key concepts present in the original text during summarization. The issue of interpretability in LLMs has been highlighted, noting that their complexity and vast size significantly complicate the understanding of their decision-making processes [63, 85]. Additionally, the occurrence of hallucinations, where the models generate inaccurate or non-

sensical information, further exacerbates these challenges. Another significant concern with LLMs is their context limitation. Over the past six years, the context size of language models has expanded dramatically, from 512 tokens in pre-trained LMs to an impressive 128,000 tokens in LLMs. However, we should also remember that, no matter how large the context size of the LLMs is increased, there can always be a situation where the input text is too large for the context size which is a critical challenge for text summarization. Infinitely increasing the context size may also initiate some drawbacks to text summarization. However, the performance of LLMs is highly dependent on the position of relevant information within the input. This means that in long prompts, information situated in the middle tends to receive less attention. This phenomenon is also known as the "lost-in-the-middle" problem [54]. This uneven distribution of attention is detrimental to summarization tasks, where ideally, every sentence, indifferent to its position in the document, should have an equal chance of being considered salient. Addressing these issues is crucial for developing more effective summarization systems.

1.3 Tree Data Structures for Text Summarization

An effective text summarization method is essential for distilling key information from larger texts into a more concise format. Though traditional methods often rely on simple string representations, they can fall short of capturing the underlying structure and relationships within the text [73]. Text data often contains hierarchical structures [96], with main ideas branching into sub-ideas, similar to how a tree has a trunk with multiple branches. Moreover, text data can include different perspectives and topics [105] which can be clustered together while summarizing the documents. Tree data structures naturally align with this hierarchical nature of the information inside text data and also the topic clusters, making them ideal for organizing and summarizing text [113]. This data structure allows for easy traversing

and data extraction, enabling us to maintain the context and relationships between different parts of the text. Tree structures are not only scalable according to the size of the input but also efficient in data retrieval, even as they grow large. They provide a clear and organized visualization of information, which is crucial for generating summaries on the go. For instance, users can prune a branch of the tree to exclude specific topics from the summary, and the summary adjusts accordingly.

In this work, we explore the use of hierarchical clustering to create these tree structures (Figure: 1.1). Hierarchical clustering [74] is a data analysis technique that groups objects based on their characteristics. For text summarization, we can use sentence embeddings to represent the characteristics of sentences. These embeddings enable us to perform hierarchical clustering on the document. There are two main approaches to hierarchical clustering: agglomerative and divisive. The agglomerative approach starts with each sentence as an individual cluster and merges the closest pairs step by step until only one cluster remains, representing the entire document. On the other hand, the divisive approach begins with the whole document as one cluster and splits it into smaller clusters progressively, illustrating the nested relationships between them [12].

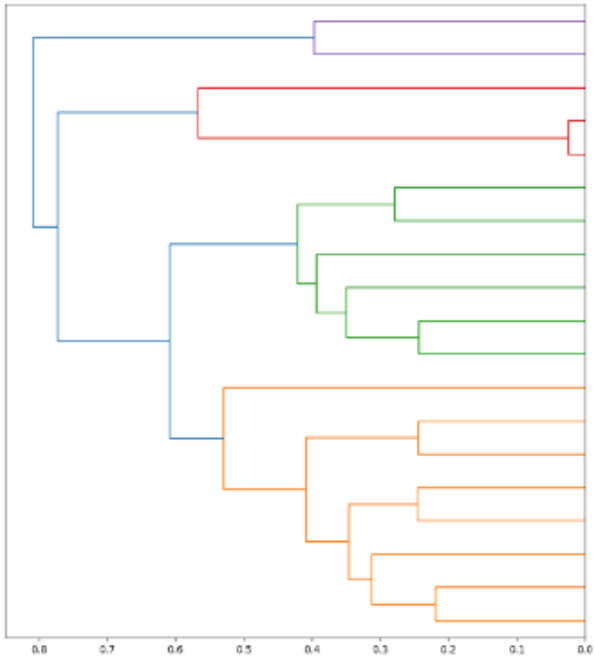


Figure 1.1: An example of a tree data structure generated with hierarchical clustering. The four colors - Orange, green, red, and purple demonstrate 4 different clusters.

One of the key advantages of hierarchical clustering is its flexibility. It does not constrain the number of clusters to a predetermined value; instead, it adapts based on cut-off points,

or thresholds, that define the clusters dynamically. This allows the hierarchy to reflect the true relationships between sentences, whether at the word, phrase, sentence, or document level. By leveraging tree data structures through hierarchical clustering, we can create more meaningful and adaptable text summaries [60]. These summaries can capture the essential structure and key points of the original text, providing a more effective tool for understanding and analyzing large volumes of information [75].

1.4 Motivation

Astro Teller, the director of Google X, remarked, "If you want cars to run at 50 miles per gallon, fine, you can retool your car a little bit. But if I told you it has to run on a gallon of gas for 500 miles, you have to start over." This analogy applies to text summarization as well. Traditional methods like abstractive and extractive summarization have inherent limitations that must be addressed. Although the advent of Large Language Models (LLMs) has rendered some abstractive techniques obsolete [78], LLMs themselves introduce new challenges [108]. Addressing these issues using the same problematic techniques is difficult. Therefore, we propose an interpretable algorithm for identifying salient sentences that also addresses most of the limitations of both the traditional and LLM-based text summarizers.

1.5 Contributions of This Thesis

Our work centers on developing a hybrid summarization system that generates coherent summaries in an interpretable manner, within the context of the given input, and with the most minimal risk of hallucination. This approach aims to mitigate the limitations of LLMs while leveraging their strengths. Our research, Docusage, contributes to the field of text summarization in three significant ways:

1. **Synthetic Summary Dataset:** We developed a synthetic summary dataset named Syn-D-sum, derived from the publicly available CNN/Dailymail (CNN/DM) dataset. This dataset serves as a valuable resource for training and evaluating summarization models, providing a robust foundation for further research and development in the field.

2. **Novel Narrative Synthesis Algorithm:** We introduce Sumsage, a novel narrative synthesis algorithm that extracts salient sentences from a given document while preserving the narrative structure of an expert human annotator. This algorithm employs hierarchical clustering to enhance the interpretability and coherence of the summaries, ensuring that the generated summaries maintain a logical flow and narrative consistency.

3. **Novel Evaluation Metric:** To better evaluate the quality of the summaries generated by Sumsage, we propose a new evaluation metric called "Symphony." This metric considers the summary's narrative structure, offering a more robust, reasonable, and intuitive scoring method compared to traditional metrics like ROUGE.

1.6 Organization

This thesis is organized as follows: Section 2 provides the background of text summarization, including the uses of LLMs and deep learning models in text summarization and narrative synthesis. Section 3 presents the algorithms we built to accomplish the task of narrative synthesis in addition to the dataset synthesis process. Section 4 demonstrates the comparison between Sumsage and GPT-3.5-turbo using the Symphony metric. Finally, Section 5 discusses the reason for the effectiveness of our algorithm and future research directions in narrative synthesis along with the limitations of the proposed system.

CHAPTER 2
LITERATURE REVIEW

Text summarization refers to the process of extracting the most important information in a source document to produce an abridged version [66]. Over the past few decades, text summarization has been an increasingly important task in Natural Language Processing (NLP) for solving different problems arising from the exponentially growing text data like information overload and scalability of analysis. Despite its importance, text summarization, is complex [91] as it requires retaining context, coherence, and critical details of the original content without losing its essence [95]. A variety of sophisticated methods have been developed to tackle the complexities of text summarization, reflecting on its significance and efforts to improve its accuracy and efficiency over time.

2.1 Early Approaches

One of the earliest methods for text summarization involved determining sentence importance based on keyword frequency [62]. In this approach, the keywords were manually identified by human annotators making them subject to human biases. Later, a comparatively more sophisticated method was proposed to rank sentences using hand-crafted features like word frequency from different parts of the document, and frequency of keywords [23]. Even though this method performed better than using only keyword frequency, using hand-crafted features still required substantial human input. These initial summarization models performed adequately given the technological limitations of the era but they required substantial human involvement. To overcome this limitation, a novel approach for automated text summarization applied a greedy algorithm to combine sentence relevance with information novelty to extract the most important sentences [13]. This research laid the foundation

for new approaches in text summarization by raising the critical question of how to automatically select the most important sentences from an input document effectively and efficiently.

In the search for an answer to that question, text summarization has been turned into different popular optimization problems such as the maximum coverage problem [89], and tree knapsack problem [34] and was approached using optimization methods such as integer linear programming (ILP) [68], discrete point process (DPP) [45], and submodular functions [53].

While deep learning methods have overshadowed optimization-based text summarizers, Cho et al. revisited DPP in 2019 to highlight its effectiveness in addressing information redundancy in multi-document text summarization [17]. Despite outperforming strong summarization baselines on benchmark datasets, they could not scale to address the current challenges presented by LLMs like hallucination, lack of interpretability, and providing users less control [108], or even other issues associated with multi-document text summarization, such as multiple viewpoints, information overload, and maintaining coherence between ideas [71].

2.2 Pre-transformer Deep Learning Approaches

Recurrent Neural Network (RNN) has been the dominant model for natural language processing (NLP) for over a decade for its capability of handling sequential data like time series and text data [24]. Though RNN-based models became widely popular in language modeling, their use was limited in text summarization because they lacked both contextual information about words and the capability to handle very long sequences, i.e., sentences [29, 21]. The task gained noticeable attention when Convolutional Neural Networks (CNN) [43], and Long Short-Term Memory (LSTM) [32] became the de facto standard neural network architectures

in natural language processing [108]. Though the performance of RNN [67], LSTM [46], and CNN [4] applied to text summarization as standalone models were not satisfactory, a combination of a feature extractor model (i.e., CNN) and a sequential model (i.e., RNN) was able to capture the nuances of language so well that it introduced a new horizon of research in this field [65]. CNN models perform well as feature extractors but cannot capture long sequential information while RNN/LSTM are not good at extracting features but can model sequential data well. Thus, by combining their strengths and mitigating their weaknesses, a more robust model was created [42]. Based on this idea, a hierarchical document encoder using CNN-RNN architecture (Also known as "Convolutional RNN") was proposed, where CNN extracted sentence-level features and RNN captured sequential information [16]. This, however, did not completely solve the problem as RNNs are susceptible to the vanishing gradient problem [35]. In neural networks, this occurs when updates to the model's weights become so small that the network becomes unable to learn and improve. On the other hand, LSTM, another type of architecture that also captures sequential information in data, is better than RNNs at handling that because its architecture includes memory cells and gating mechanisms that regulate the flow of information and gradients over long sequences [41]. Hence the RNN part of the CNN-RNN model was replaced with an LSTM network to upgrade it to a CNN-LSTM model architecture (Also known as "Convolutional LSTM") [84, 86]. Later, the attention mechanism [6] was introduced as a better method to capture the context of words in a document than CNNs [77]. Therefore, the CNN part of the CNN-LSTM sequence-to-sequence model was replaced with the attention mechanism, which led to a more robust model called the Attention-LSTM model [33, 40, 102]. By utilizing the power of this architecture, the Discourse-Aware Attention Model [19] was introduced which uses a combination of two bi-directional LSTMs and the attention mechanism for high-quality text summarization. Building on top of the idea, a more robust and powerful extractive summarization model was proposed which incorporates the attention of different parts of a

long document into a Discourse-Aware Attention Model [101].

2.3 Transformer Models

The introduction of transformer models, which are better at capturing long-range dependencies with self-attention mechanisms, significantly improved the summarization performance compared to RNNs and LSTMs [56]. Transformer-based models like Bidirectional Encoder Representations from Transformers (BERT) [22] paved the way for significant advancements in various natural language processing (NLP) tasks by introducing transfer learning, enhancing contextual understanding, and standardization of pre-training techniques. Similarly, generative models like open-AI's decoder-only models with generative pre-training (GPT) [11] demonstrate the potential of large-scale unsupervised learning for text generation, leading to the development of modern large language models (LLMs).

Pretraining language models gained popularity with the introduction of BERT. Subsequently, numerous BERT-based models fine-tuned for specific tasks have demonstrated significant performance improvements, advancing the capabilities of natural language processing [93, 49, 61]. This has led to the development and utilization of a variety of pre-trained models for text summarization, each fine-tuned to excel in generating concise and coherent summaries across different types of text and contexts such as Pegasus [109], BART [50], and T5 [79]. SimCLS [59] harnessed the generative strengths of BART to produce diverse and contextually appropriate summaries while relying on RoBERTa's robust evaluation capabilities [57] to ensure the selection of the highest quality summaries.

Even though these pre-trained models have a deep contextual understanding of words and sentences, they cannot effectively capture and utilize the relationships between words, phrases, sentences, and the overall document. This limitation led to the proposal of the hierarchical transformer model [55] for text summarization that builds a vector representation

of a document by considering the relationships between different units of natural language like words, phrases, and sentences. They combine word-level representations at the bottom and gradually incorporate phrase and sentence-level vector representations to construct a document-level embedding. The embeddings generated by this algorithm are used in our research as they provide a robust representation of both the sentences and the document. One of the limitations of the model is that due to the hierarchical structure of the document embeddings and the sentence embeddings, they are very close to each other in the embedding space. Therefore, we normalize the cosine similarities between the document and the sentences on a scale of 0 to 1 to achieve a robust scoring system.

2.4 Large Language Models

A recent trend in natural language processing has been the dominance of Large Language Models (LLMs), specifically variations of GPT, due to their exceptional ability to understand and generate natural language. They have set a new standard in NLP by processing and generating coherent, and contextually relevant text [47], redefining to what extent machines are capable of comprehending human speech. Therefore, increased attention was given to exploring and enhancing LLM capabilities, seeking to harness their full potential in diverse linguistic tasks such as question answering [81], machine translation [81], and text summarization [37].

The majority of current studies on the use of LLMs in text summarization focus on their performance across a variety of summarization tasks as well as their attributes and behavior [111, 30, 51, 36, 106]. While the initial LLMs were exceptional at producing natural language, they did not possess the same level of reasoning abilities [14]. Thus, when the chain of thought (CoT) technique was introduced to the field of LLMs for text summarization, research in the field was invigorated. It enhanced LLM's reasoning abilities and facilitated better summaries

[97]. When summarizing text, extensive prompting techniques such as CoT require longer context. As a solution, Chang et. al. [15] introduced a method to hierarchically merge chunk-level summaries focusing on the use of LLMs for book-length summaries.

Observing the performance and rapid growth of LLMs in text summarization, it has been proclaimed that the task is "almost dead" [78]. However, subsequent research revealed significant limitations to an LLM-based summarizer. There are still open challenges in the development of LLM-based text generators relating to issues like hallucination, fairness, and bias [112, 90]. Moreover, the research community has also failed to adequately address important issues such as the interpretability and explainability of LLMs [85, 10, 25, 64]. We are still in the early stages of maximizing LLMs' potential for summarization despite considerable research efforts directed at this task [88, 1, 107, 58].

2.5 Evaluation

Despite the ability of large language models (LLMs) to pass the Turing test [38], LLM-based text summarizers are yet to achieve leading performance on benchmark tests. The state-of-the-art model for abstractive text summarization is a reinforcement learning-based model evaluated using the ROUGE metric [98]. This model, known as Reinforced Neural Extractive Summarization (RNES), employs a neural extractive summarizer that is rewarded for maintaining high inter-sentence coherence and achieving a high ROUGE score. A major focus of the approach is the improvement of ROUGE scores through reinforcement learning. However, high ROUGE scores do not necessarily ensure high-quality summaries [2]. According to Mu et al. [72], metrics such as ROUGE, METEOR [7], and BERTScore [110] can be attacked to achieve higher scores even when summaries are of inferior quality. Despite working with non-summary strings, their attacker model produced high ROUGE and METEOR scores competing with the top summarizers. This raised concerns regarding the robustness

and reliability of current automated evaluation metrics and underscored the need for a new scoring mechanism. These concerns have led researchers to explore alternative evaluation criteria beyond traditional metrics. Current research emphasizes the importance of assessing whether summaries adhere to a good narrative structure. This involves identifying which sentences are considered important and determining the optimal sequence of these sentences to create an effective story flow [110]. By focusing on narrative coherence and the logical arrangement of information, researchers aim to develop more reliable evaluation methods that reflect true summary quality [48, 87]. This shift in focus highlights the evolving understanding of what constitutes a high-quality summary and the need for more comprehensive assessment tools. There are several methods proposed to evaluate summaries, such as, using coherence error detection [31] or movements in embedding space [114, 18]. However, these methods are unable to address the narrative sentence structure or other significant aspects of summaries such as relevance and fluency. UniEval was one of the very few metrics to incorporate coherence, consistency, fluency, and relevance into a single scoring system [115]. Their approach converts the evaluation problem into a boolean question-answer problem, which necessitates a different set of training data. UniEval relies on external knowledge from multiple related tasks to generate its scores, increasing its computational cost and reducing its consistency and reliability. Therefore, as text summarization continues to evolve with more sophisticated models and evaluation techniques, a comprehensive generation and evaluation framework that integrates coherence, consistency, fluency, and relevance while addressing the current challenges is essential for advancing the quality of summarization models.

CHAPTER 3
METHODS

3.1 Information Backtracking Algorithm (IBA)

Summarization involves selecting key information from a few sentences rather than including all sentences from the input document [66]. This raises a critical research question: how should key sentences be selected? The Information Backtracking Algorithm aims to identify sentences that contribute the most information to summaries while avoiding redundancy and remaining concise. Summaries do not necessarily contain all the information from the input document but only the most important details (Figure 3.1). This leads to another crucial question: which information should be considered important in a document? Here, we define important information as those which a human expert in the field decided to include in. To identify the source of information in human-written summaries, we employ the Information Backtracking Algorithm (IBA).

Sample Document: Document In a significant advancement, researchers at the University of California San Francisco, have developed a new, cost-effective battery technology that doubles the energy storage capacity of existing solutions. This innovation, announced today, uses environmentally friendly materials and promises to enhance the efficiency of renewable energy systems by enabling longer storage of solar and wind energy. The new battery is expected to hit the market by late 2025, potentially revolutionizing energy grids worldwide.

Sample summary: Researchers at the University of California have developed a new, eco-friendly battery technology that doubles the storage capacity of renewable energy.

Figure 3.1: Example showing the source of information in the summary. The information is color-coded to match the phrases from the summary with their corresponding sources in the document.

Here we use the concept of embeddings [70], which convert text into numerical vectors by encoding their meanings and contexts. Cosine similarity measures the relatedness of two vectors, with higher values indicating greater similarity. Thus, sentences with high similarity

to a specific word or phrase in a summary can be considered sources of that word or phrase. Therefore, by calculating cosine similarity, we can trace each word or phrase in the summary back to its most relevant sentence(s) in the input text. For example, in Figure 3.1, the first sentence is expected to have high cosine similarity with the phrase "Researchers at the University of California" due to their similar context and content. By applying this process to human-written summaries, we determine which sentences expert annotators focus on while writing them. This method is referred to as the IBA, as it involves backtracking the information in summaries to find their sources in the input document. A step-by-step workflow of IBA is as follows:

Embedding Generation: The first step in IBA involves tokenizing both the input text and the summary into individual words or tokens. Each token is then converted into a numerical vector representation using word embeddings. These embeddings encode the meanings and contexts of words into numerical vectors, facilitating subsequent similarity calculations.

Similarity Match: For each word or phrase in the summary, we calculate the cosine similarity between its embedding and the embeddings of words in the input text. Cosine similarity ranges from -1 (complete dissimilarity) to 1 (complete similarity).

Source Identification: For each token in the summary, IBA identifies the sentence(s) in the input text with the highest cosine similarity. These sentences are considered the primary sources of the information contained in the summary.

Algorithm 1 provides a comprehensive overview of the entire procedure employed to backtrack information to their sources (Figure 3.2).

Algorithm 1: INFORMATION BACKTRACKING ALGORITHM

Input: Q = query document, DB = database document, T = similarity threshold, k = number of top sentences, model = pre-trained word embedding model

Output: $SalientSentences$ = list of salient sentences for each query word

for $q \in Q$ **do**
    split q into words;
    generate embeddings for each word in q using model;
for $d \in DB$ **do**
    split d into words;
    generate embeddings for each word in d using model;
for $q \in Q$ **do**
    for $Q_{emb_{SiWj}} \in q$ **do**
        $highest\_similarity \leftarrow 0$;
        $sources \leftarrow [\,]$;
        for $DB_{emb_{SxWy}} \in DB$ **do**
            $sim \leftarrow calculate\_similarity(Q_{emb_{SiWj}}, DB_{emb_{SxWy}})$;
            if $sim > highest\_similarity$ **then**
                $highest\_similarity \leftarrow sim$;
        for $DB_{emb_{SxWy}} \in DB$ **do**
            $sim \leftarrow calculate\_similarity(Q_{emb_{SiWj}}, DB_{emb_{SxWy}})$;
            if $highest\_similarity - sim < T$ **then**
                add $DB_{emb_{SxWy}}$ to $sources$;
    $matched\_words\_count \leftarrow [\,]$;
    for $s \in DB$ **do**
        $match\_count \leftarrow 0$;
        for $w \in s$ **do**
            if $w \in sources$ **then**
                $match\_count \leftarrow match\_count + 1$;
        $matched\_words\_count \leftarrow matched\_words\_count \cup \{match\_count\}$;
    $salient\_sentences \leftarrow$ identify top k sentences from $matched\_words\_count$;
    add $salient\_sentences$ to $SalientSentences$;
return $SalientSentences$;

3.2 Dataset Synthesis

To align the behavior of our summarizer with that of human writers, it is essential to first determine which sentences in an input document are deemed important by humans. This requires a dataset containing source sentences that expert humans have identified as important for creating summaries. Currently, no public dataset exists that provides these specific source sentences considered important in human-written summaries. Therefore, we create

a synthetic dataset called "SYNthetic Data for text SUMmarization" (Syn-D-Sum) to facilitate the development and evaluation of text summarization algorithms by providing a benchmark dataset that extracts the most salient sentences in documents, as identified by expert annotators. We selected the CNN/DM public dataset as the seed dataset for Syn-D-Sum due to its diverse news articles and highlights written by human experts. This dataset was preprocessed to remove advertisements and unrelated content. The news articles (referred to as "documents" in the paper) contain an average of 751 tokens and the highlights (referred to as "summaries" in the paper) contain an average of 56 tokens.

Figure 3.1 illustrates how various segments of sentences within an example document contribute to the information in the corresponding summary. Our algorithm is designed to extract the information encapsulated in the summaries written by expert humans. This necessitates a dataset that identifies which sentences in a document contribute to the information present in the summaries, or in other words, are salient.

To address this, we employ a salient sentence identification algorithm known as the Information Backtracking Algorithm (IBA). This algorithm enables us to identify the salient sentences that are integral to the information present in a human-written summary, ensuring that our system accurately captures the most relevant content.

Figure 3.2 shows a flow diagram of IBA. Initially, we tokenize each document and its corresponding summary into sentences, and subsequently, each sentence into words. For both the document and the summary, we generate an embedding for each word. Given a set of sentences, $S = \{s_1, s_2, \ldots, s_n\}$ and a set of words from the summary, $W = \{w_1, w_2, \ldots, w_m\}$, we perform a soft-match for each word w_i across all the words in each sentence within the set S. This matching process is based on the word embeddings of all the words in each sentence of the document and the summary. The word embeddings were calculated using the bge-large-en-v1.5 [100] embedding model, which was the state-of-the-art model for embeddings at the time of dataset synthesis.

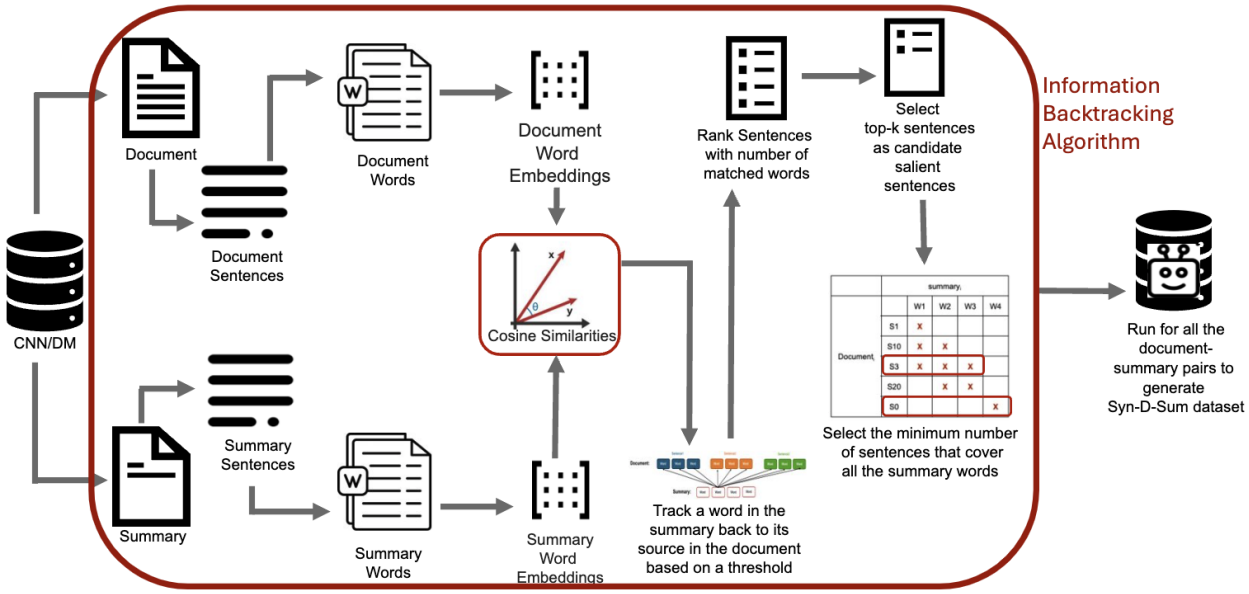


Figure 3.2: A high-level flow diagram of the Dataset Synthesis Algorithm showing each step taken to synthesize Syn-D-Sum from CNN/DM.

We count the number of words in each sentence, s_i that matches each word in set W with a function M as follows.

$$M(s_i, W) = \sum_{w \in W} \delta(w \in s_i),$$

where δ is an indicator function that returns 1 if w is in s_i, and 0 otherwise. Afterward, we extract the top 5 sentences with the highest number of matched words.

Subsequently, we select the top k sentences by sorting them primarily by the number of unique words they cover, and secondarily by the total number of words covered.

$$\text{Top-k}(S, W) = \{s_j \in S \mid j \in \arg \text{ top-k}(M(s_i, W))\},$$

where arg top-k$(M(s_i, W))$ represents the indices of the top k values of $M(s_i, W)$. In our research, we used k=5 to extract the top 5 sentences. These selected sentences are marked as salient sentences for that specific document and are ordered as they appear in the summary.

This process is repeated for 12,798 document-summary pairs from the validation data split of the CNN/DM dataset, resulting in the creation of our Syn-D-Sum dataset. Syn-D-Sum represents the most salient sentences in a document, arranged in the same order as they appear in the human-written summaries. This entire procedure, as depicted in Algorithm 1, provides a comprehensive overview of the process employed to generate the Syn-D-Sum dataset.

These selected sentences are marked as salient sentences for that specific document and are ordered as they appear in the summary. This process is repeated for 12,798 document-summary pairs from the validation data split of the CNN/DM dataset, resulting in the creation of our Syn-D-Sum dataset. Syn-D-Sum represents the most salient sentences in a document, arranged in the same order as they appear in the human-written summaries.

	$summary_i$				
$Document_i$		W1	W2	W3	W4
	S1	X			
	S10	X	X		
	S3	X	X	X	
	S20		X	X	
	S0				X

Figure 3.3: A sample result of the Information Backtracking Algorithm (IBA) showing matches between document sentences and summary words. Sentences S3 and S0 together cover all summary words (W1, W2, W3, W4), making them the selected salient sentences.

Figure 3.3 illustrates a dummy result of IBA to explain the process more clearly. In this example, words W1 and W2 of a summary sentence match with three sentences in the document each, W2 matches with two, and W4 matches with one. Evaluating the results per row, it is observed that S1 in the document covers W1 of the summary, S10 covers both W1 and W2 and so on. This implies that by selecting S3, which covers information from three words in the summary sentence, and S0, which uniquely covers the information in the fourth word, we can encompass all the information present in the summary sentence with just two sentences from the document. Therefore, we select sentence number 3 and 0 as salient sentences.

3.3 Sumsage Algorithm

The Sumsage algorithm leverages hierarchical clustering to generate robust and concise narratives from documents. This process begins by generating document embeddings for each document and sentence embeddings for each sentence within the document, utilizing a hierarchical transformer embedding model [55].

By calculating the inter-sentence cosine distances, a hierarchical clustering tree is constructed from the embeddings of all the sentences in the document. Subsequently, an importance score is assigned to each sentence based on its cosine similarity with the document embedding, referred to as the sentence's "thickness" value (Figure 3.4). The design of the Sumsage algorithm generates varying narratives depending on different starting points. Therefore, for selecting the initial sentence in the narrative, the algorithm selects the first sentence from the gold standard reference and iteratively attempts to converge on the same narrative structure as the human-written summaries. This approach ensures that the generated narrative closely follows the logical and coherent structure identified by human experts.

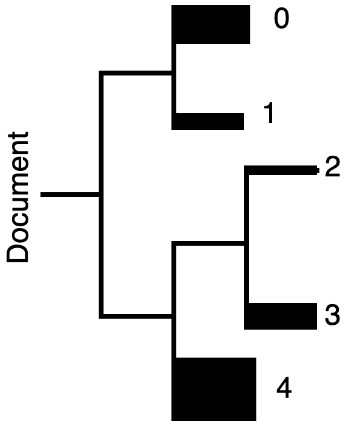


Figure 3.4: A sample tree structure with thickness values (cosine similarity with the document embedding) of each sentence.

Figure 3.5 shows a high-level flow diagram of the process we used in order to generate the narratives. Algorithm 2 also depicts the iterative steps taken to generate narratives, ensuring it aligns closely with the gold standard in terms of structure and content. This rigorous approach aims to produce narratives that are not only concise and relevant but also logically coherent and reflective of human-written summaries.

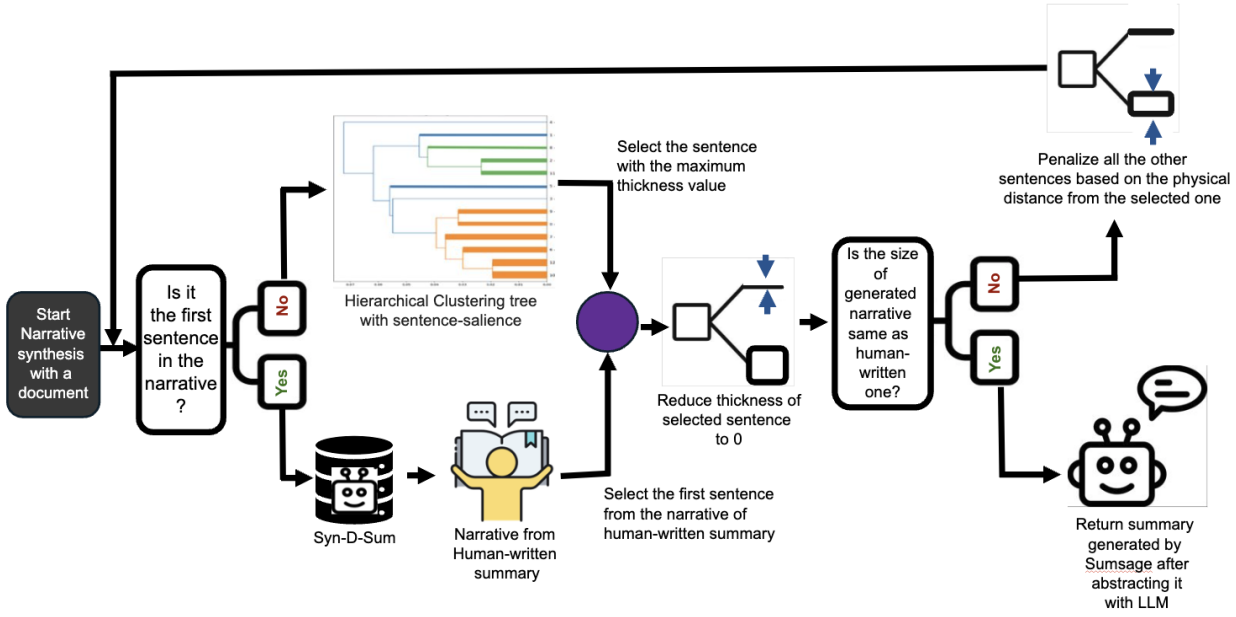


Figure 3.5: A high-level flow diagram of the Sumsage Algorithm showing the steps taken to generate the narratives.

Algorithm 2: SUMSAGE SUMMARIZATION ALGORITHM

Input: D = set of documents, G = gold standard narratives from Syn-D-Sum dataset
Output: $Narrative$ = generated narrative for each document

```
for d ∈ D do
    Extract gold standard narrative g for document d from G;
    while Size(Narrative) ≠ Size(g) do
        if first iteration then
            Pick the first sentence from g as the starting sentence;
        else
            Pick the thickest sentence s from d;
        Add s to Narrative;
        Set the thickness of the picked sentence s to 0;
        for each remaining sentence r in d do
            Penalize the thickness of r according to the penalty function;
return Narrative;
```

We set the thickness value of the first sentence to 0 to stop it from being considered for a summary sentence again and penalize all the other sentences based on the penalization function shown in Equation 3.1.

$$penaltyfunction = \left(\frac{ae^{(ua)} + be^{(ub)}}{e^{(ua)} + e^{(ub)}} \right) \tag{3.1}$$

where the penalty function is the Boltzmann max of two functions a and b. Here a and b are the parameterized versions of the reverse sigmoid function (equation 3.2) and the sigmoid function (equation 3.3) respectively.

$$a = \frac{1}{1 + e^{-(ix - m_1)}} \tag{3.2}$$

$$b = \frac{1}{1 + e^{(jx - m_2)}} \tag{3.3}$$

Taking the Boltzmann maximum approximation of the two functions results in a convex and smooth penalty function which addresses our objective of selecting sentences for the narrative that are not redundant and also do not jump back and forth between topics (Figure 3.6).

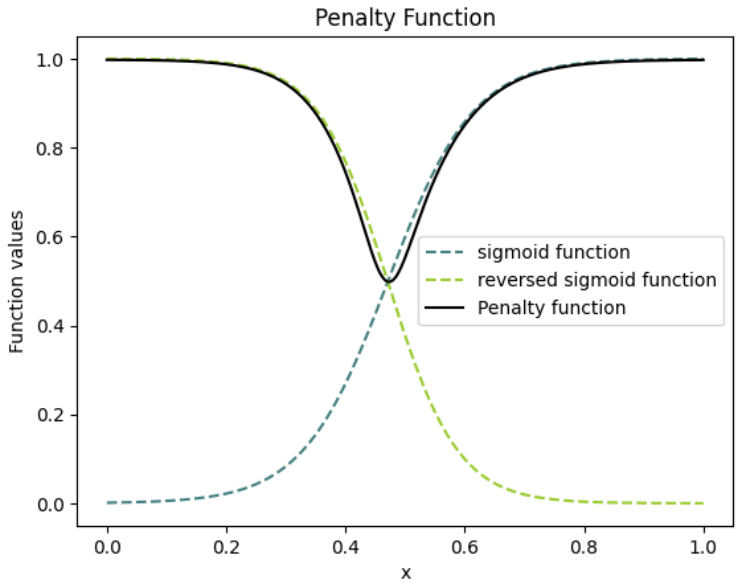


Figure 3.6: The penalty function is the Boltzmann Max of a parameterized sigmoid function and a reverse sigmoid function.

The penalty function depends on the physical distance between the embeddings of the selected sentence and each of the other sentences in the hierarchical clustering tree. It

heavily penalizes sentences that are very distant from the selected sentences to avoid the narrative jumping back and forth between topics very often and very close to them to avoid the narrative being stuck on the same topic. It assigns a smaller penalty to sentences that are at an intermediate distance from the selected sentence. After penalizing all other sentences, the most important sentence is chosen, and its thickness is set to 0 to prevent it from being selected again in the narrative construction. This process is repeated until the constructed narrative matches the length of the gold standard narrative.

We frame our narrative selection algorithm as a hyperparameter optimization problem for the penalty function. The parameters of the penalty function are initially set to values that yield the highest accuracy when processed through a Bayesian optimizer. Upon generating a narrative for each document, we further optimize the hyperparameters of our penalty function based on a loss function designed to minimize the total pairwise distance between the gold standard and the generated narratives.

For a set of gold standard narratives, $G = \{G1, G2, \ldots, Gn\}$ and a set of generated narratives $N = \{N1, N2, \ldots, Nn\}$. The objective is to find the optimal parameters θ that minimize the total pairwise distance:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} d(G_i, N_i) \tag{3.4}$$

Since the generated narratives N_i are influenced by the penalty function P, we can express N_i as a function of θ:

$$N_i = f(P(N_i, \theta)) \tag{3.5}$$

Therefore, by combining equation 3.4 and equation 3.5 the optimization problem can be reformulated as

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} d(G_i, f(P(N_i, \theta))), \tag{3.6}$$

where θ are the hyperparameters of the penalty function and f represents the generation function influenced by the penalty. This formulation represents the optimization of the hyperparameters of the penalty function to minimize the total pairwise distance between the gold standard and the generated narratives. Upon convergence, we use the optimized hyperparameters to generate the narratives from 4798 test documents.

Our approach employs an extract-then-abstract summarization model, leveraging the capabilities of LLMs to produce coherent and comprehensive summaries of input documents. The method identifies and extracts key narrative sentences that follow the narrative of the human-written summaries. Once these sentences are selected, the LLM performs a controlled abstraction process. This process involves synthesizing the extracted information into a cohesive summary that not only preserves the essential points but also enhances the overall readability and flow. By carefully abstracting the content, the LLM ensures that the summary is both concise and reflective of the original document's intent and nuances. This method effectively balances the necessity of detail retention to produce a clear, understandable summary, making it highly suitable for a wide range of applications in both academic and professional contexts.

3.4 Experimental Setup

We conducted gradient-based optimization on 8000 documents from the Syn-D-sum dataset, derived from the validation set of the public CNN/DM dataset. The optimization process included an early stopping condition, which halted the process if the validation loss on a set of 1000 validation data points did not decrease for 5 consecutive iterations. The optimization was performed using both the Stochastic Gradient Descent (SGD) [80] and

Adam [44] optimizers separately, each on a single Tesla A6000 GPU. Results showed that the SGD optimizer yielded higher accuracy, whereas the Adam optimizer converged about 7 times faster. For the subsequent experiments, we utilized the hyperparameters obtained by optimizing the penalty function with the SGD optimizer.

CHAPTER 4
EVALUATION

The most popular evaluation metrics for reporting text summarization tasks are the ROUGE-class of metrics for their ability to gauge the order of content in summaries [27]. The most popular ROUGE metric for text summarization, the ROUGE-L metric [52], is calculated based on the longest common subsequence of words. Despite its widespread popularity, ROUGE metrics are inadequate for evaluating summarized narratives generated by Sumsage because they evaluate summaries based on n-gram matches, and penalize models that generate new wordings and phrases [2]. This focus on exact word matching means that summaries using different but equally valid expressions are scored lower. As a result, the ROUGE-class of methods fails to recognize creative and diverse ways in which the same content can be summarized. Furthermore, ROUGE's reliance on n-gram overlap does not account for the coherence and readability of the summary. A summary with high n-gram overlap might still be poorly structured or difficult to understand. The ROUGE-class of methods overlooks these aspects, which are crucial for evaluating the effectiveness of a summary. To address these issues, we propose a new evaluation penalty - Symphony, which evaluates whether a generated narrative follows the gold standard narrative in terms of matching and ordering of important sentences, and an evaluation score - Captured Importance Quantification (CIQ), which evaluates the importance captured by the sentences in the generated narrative.

4.1 Novel Evaluation Penalty: Symphony

Symphony considers the diversity, coherence, and overall quality of the summaries which makes it a reasonable evaluation penalty for evaluating the summaries generated by Sumsage. Symphony evaluates two main aspects of narrative generation.

a. **Matching**: Does the summary select from the original text the same sentences as those appearing in the gold standard?

b. **Ordering**: Do the sentences selected for inclusion in the summary appear in the same order as in the gold standard?

These two objectives make sure that the generated narrative closely follows the human-generated narratives, i.e., the gold standard. Moreover, the embedding-based objectives ensure a more robust way to evaluate how similar the generated narrative is to the gold standard in comparison to the Longest Common Subsequence (LCS) matching of the ROUGE-L metric.

4.1.1 Symphony Evaluation Method

To evaluate the matching objective (a), we calculate a *matching penalty* based on how far each sentence in the generated narrative is from the closest match in the gold standard using their distances in the hierarchical clustering tree. The *matching penalty* is calculated using the following equation

$$\text{matching penalty} = \frac{1}{n}\sum_{i=1}^{n} d(\text{gold}_i, \text{gen}_{\text{closest}}), \tag{4.1}$$

where gold_i is each sentence in the gold standard narrative and $\text{gen}_{\text{closest}}$ is the sentence in the generated narrative closest to the gold standard sentence.

The *ordering penalty* is calculated based on the total pairwise distance between the gold standard and the generated narratives. For a set of sentences in the gold standard narrative, $\text{GOL} = \{\text{gol}_1, \text{gol}_2, \text{gol}_3, \ldots, \text{gol}_n\}$ and a set of sentences in the generated narrative, $\text{GEN} = \{\text{gen}_1, \text{gen}_2, \text{gen}_3, \ldots, \text{gen}_n\}$.

The total pairwise distance of the gold standard narrative is

$$d_{\text{GOL}} = \sum_{i=1}^{n-1} d(\text{gol}_i, \text{gol}_{i+1}),$$

and the total pairwise distance of the generated narrative is

$$d_{\text{GEN}} = \sum_{i=1}^{n-1} d(\text{gen}_i, \text{gen}_{i+1}),$$

where d(x,y) represents the distance between the embeddings of x and y in the hierarchical clustering tree. The *ordering penalty* is the difference between the two pairwise total distances.

$$\text{ordering penalty} = |d_{\text{GOL}} - d_{\text{GEN}}| \tag{4.2}$$

If *ordering penalty* ≈ 0, we consider the ordering of the sentences in the generated narrative a good match with that of the gold standard narrative. Upon calculating the two different penalties in equation 4.1 and equation 4.2, we aggregate them using a Weighted Quadratic Mean function(equation 4.4) to get the Symphony penalty of the generated narratives. The equation for the Symphony penalty is as follows

$$\text{Symphony penalty} = \text{QMean}_w((\text{matching penalty}, 0.6), (\text{ordering penalty}, 0.4)) \tag{4.3}$$

where the QMean_w function is calculated with the following equation

$$\text{QMean}_w = \sqrt{\frac{\sum_{i=1}^{n} w_i x_i^2}{\sum_{i=1}^{n} w_i}} \tag{4.4}$$

Here w_i is the weight for each parameter passed into the QMean_w function and x_i is each penalty we are aggregating. We decided to weight the *matching penalty* by 0.6 and

the *ordering penalty* by 0.4 as we want our evaluation penalty to focus more on matching the sentences than their order. We experimented with different combinations of weights on *matching penalty* and *ordering penalty* and found the best results when setting them at 0.6 and 0.4 respectively. This assigns a slightly increased weight to matching the important sentences rather than ordering them correctly. However, the weights can vary based on different objectives which makes the Symphony penalty adaptive to different use cases.

4.1.2 Symphony Evaluation Result

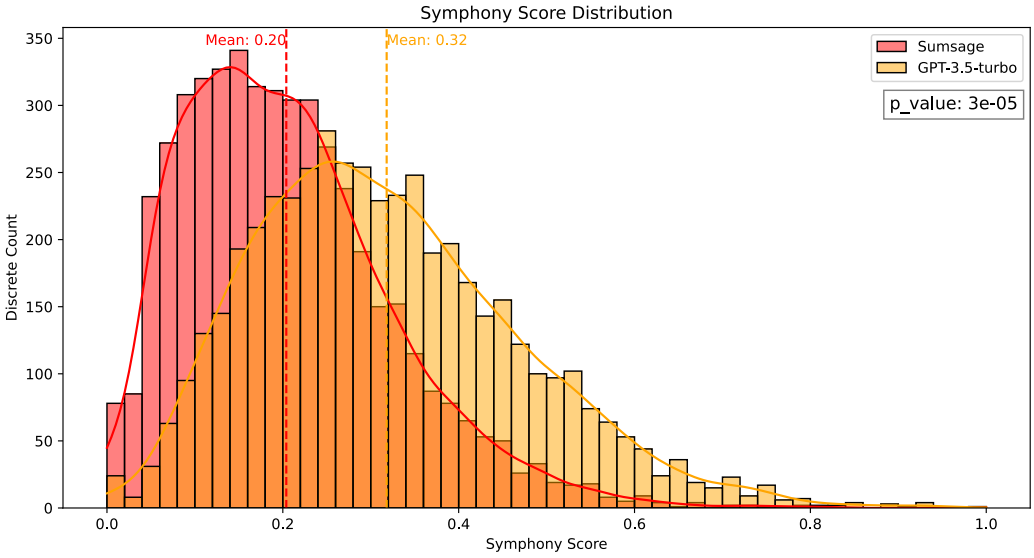


Figure 4.1: Comparative distribution of the "Symphony penalties" between the Sumsage method and GPT-3.5-Turbo (lower penalties are better).

We compute the symphony penalty for 4798 document summaries generated using Sumsage and GPT-3.5-turbo. The distribution of the resulting Symphony penalties are shown in Figure 4.1, where the dotted lines show the mean of the penalty distribution for each summarization method. A Mann-Whitney U test [69] on the 2 distributions of Sumsage and GPT-3.5-turbo and get a p-value of 0.00003 (< 0.05) that indicates a statistically significant

Example 1: Sumsage performed better than GPT-3.5-turbo.

Document: france will look to inject some life into a miserable six nations campaign thus far as philippe saint-andre travel to rome to face italy on sunday. les bleus completed their final preparations at their team base in marcoussis ahead of their clash with the azzurri at the stadio olimpico. under-fire france boss saint-andre has rung the changes following his sides 20-13 loss to wales in paris last time out. france lock yoann maestri (centre) makes a run as hooker benjamin kayser (left) watches on. castres scrum half rory kockott (centre) and prop eddy ben arous (second left) listen in at training. france captain thierry dusautoir (left) and clermont prop vincent debaty get through some sled work. young toulouse centre gael fickou (left) has been recalled to frances stating line-up. france head coach philippe saint-andre watches on as his side limber up at their team base south of paris. clermont winger noa nakaitaci. uni antonio, romain taoffifenua, scott spedding and fly-half jules plisson arrive for a training. mathieu bastareaud (left) runs with a ball as hooker guilhem guirado tries to half the centres progress. clermont winger noa nakaitaci becomes the 82nd player to be used by saint-andre; the same number that previous boss marc lievremont used during his oft-criticised four-year stint at the helm of the french team. nakaitaci is one of eight changes to the starting line-up with scott spedding, gael fickou, maxime mermoz, sébastian tillous-borde, loann goujon, alexandre flanquart and nicolas mas all earning recalls.

Gold standard summary:

Philippe Saint-André 's side lost 20-13 to Wales in the previous round. Saint-Andre has made eight changes to his starting line-up. Noa Nakaitaci becomes the 82nd player to be used by Saint-André. Gael Fickou and Maxime Mermoz form a new centre partnership.

Sumsage After Abstraction with LLM:

Under-fire France boss Philippe Saint-André has made substantial changes to the team following France's defeat by 20-13 to Wales in Paris. Clermont winger Noa Nakaitaci has been included, becoming the 82nd player used by Saint-André. This matches the number of players utilized by his predecessor, Marc Lièvremont, during his much-criticized four-year tenure. Nakaitaci is one of eight new additions to the starting line-up. Alongside him, Scott Spedding, Gael Fickou , Maxime Mermoz , Sébastian Tillous-Borde, Loann Goujon, Alexandre Flanquart, and Nicolas Mas have all been recalled. Young Toulouse centre Gael Fickou has also been reinstated to the starting line-up, indicating a fresh approach after the disappointing performance against Wales.

GPT-3.5-Turbo:

France is looking to improve their performance in the Six Nations campaign as they prepare to face Italy in Rome. Under-fire coach Philippe Saint-Andre has made eight changes to the starting line-up, including the recall of players such as Gael Fickou and Scott Spedding. Clermont winger Noa Nakaitaci is one of the new additions to the team, becoming the 82nd player to be used by Saint-Andre. The team has completed their final preparations at their base in Marcoussis ahead of the match.

Example 2: GPT-3.5-turbo performed better than Sumsage.

Document: cardinal edward egan, who led the archdiocese of new york for nearly a decade, died thursday at 82, the archdiocese announced. egan was pronounced dead of a heart attack at nyu langone medical center at 2:20 p.m. "thank god he had a peaceful death, passing away right after lunch today," cardinal timothy dolan, egan's successor and current archbishop of new york, said in a statement. egan was appointed archbishop of new york in 2000 and later cardinal in 2001 by pope john paul ii. egan retired from the archdiocese in 2009. as the leader of one of the largest catholic communities in the country, egan oversaw a growth of more than 200,000 registered parishioners and welcomed pope benedict xvi on a visit to new york city in 2008 to commemorate the 200th anniversary of the founding of the diocese of new york. "cardinal egan spread love and knowledge, and brought comfort to countless new yorkers and others across the country and the world who sought his guidance and counsel – especially in the aftermath of 9/11," said new york mayor bill de blasio in a statement. egan served previously as the bishop of bridgeport, connecticut, from 1988 to 2000, when the diocese was at the center of a child sex abuse scandal. the bridgeport diocese in 2001 paid $15 million to 26 plaintiffs to settle sexual abuse claims against eight priests. egan apologized for his involvement in the scandal in a letter read across the new york archdiocese in 2002, though he later retracted that apology in an interview with connecticut magazine 10 years later. "i should never have said that," egan said. "i did say 'if we did anything wrong, i'm sorry,' but i don't think we did anything wrong." egan was born in oak park, illinois, in 1932 and was ordained in rome at the pontifical north american college in 1957. funeral arrangements are pending, the archdiocese said. people we've lost in 2015 .'

Gold standard summary:

cardinal edward egan died of a heart attack . he served as archbishop of new york from 2000 to 2009 .

Sumsage After Abstraction with LLM:

Cardinal Edward Egan , who led the Archdiocese of New York for nearly a decade, died on Thursday at the age of 82, the Archdiocese announced. Egan retired from the Archdiocese in 2009 as the leader of one of the largest Catholic communities in the country. During his tenure, he oversaw the growth of 200,000 registered parishioners and welcomed Pope Benedict XVI to New York City in 2008 to commemorate the 200th anniversary of the founding of the Diocese of New York.

GPT-3.5-Turbo:

cardinal edward egan , former archbishop of new york , passed away at the age of 82 from a heart attack . he served as archbishop from 2000 to 2009 and was appointed cardinal in 2001 by pope john paul ii. egan oversaw a growth in the catholic community during his time in new york and welcomed pope benedict xvi in 2008. he also previously served as the bishop of bridgeport, connecticut, during a child sex abuse scandal. egan's involvement in the scandal led to a settlement of $15 million in 2001. funeral arrangements for egan are pending.

difference between the two distributions.

We further analyze two different cases where in one example Sumsage outperforms GPT-3.5-turbo and in another example, we have the contrary. Example 1 shows how the Sumsage algorithm performs compared to the gold standard and GPT-3.5-Turbo model. Information highlighted in yellow represents key points in the gold standard summary and shows where the gold standard and the subsequent two methods overlap.

The comparison presented in example 1 shows that the summary generated by Sumsage includes all the important sentences as the gold standard, whereas the GPT-3.5-Turbo model does not consider the sentences that introduce the topic of France losing to Wales 20-13. Although Sumsage needs improvement in conciseness, it successfully captures all the key information deemed important by human annotators, which helped it achieve a perfect Symphony penalty (0.0) while GPT-3.5-Turbo received a penalty of 0.29.

In example 2, the summary generated by GPT-3.5-Turbo resulted in a Symphony penalty of 0.0, while the Sumsage-generated summary was penalized 0.06. It is evident from the yellow-marked information that GPT-3.5-Turbo deemed the same sentences important as the human annotator, whereas Sumsage identified a different set of sentences as important. This highlights the differences in the algorithms' approaches to sentence selection and importance evaluation, where GPT-3.5-Turbo aligned more closely with human judgment in this instance.

4.2 Captured Importance Quantification (CIQ)

One limitation of the Symphony scoring system is that Sumsage-generated narratives can exhibit low Symphony penalties (a desired outcome) while potentially omitting some of the most critical sentences. To address this, we conducted an additional evaluation of the Sumsage algorithm by computing the Captured Importance Quantification (CIQ) score for

the generated narratives.

4.2.1 CIQ scoring method

We utilized the thickness score, as detailed in Section 3, to serve as the "importance score" for each sentence in the gold standard narrative, calibrated on a scale of 100.

$$\sum_{g \in G} T(g) = 100,$$

where G is the set of sentences in the gold standard narrative, $T(g)$ is the scaled thickness score (importance score) for sentence $g \in G$.

It is essential to recognize that sentences within a narrative do not uniformly contribute to the overall informational content of the summary. By utilizing the thickness values of the sentences, we quantified the importance of each sentence in the gold standard narrative. Subsequently, we assessed the extent of informational overlap between the generated narratives and the gold standard narratives. This approach enables a more nuanced evaluation of the Sumsage algorithm's performance in capturing and conveying the most pertinent information from the source material.

$$\text{CIQ}(S, G) = \sum_{g \in O} T(g), \tag{4.5}$$

where O is the overlap set between the gold standard sentences and the generated narrative.

The case of model A resulting in a higher CIQ score than model B indicates that the model is considering more important sentences based on gold standard summaries as well as selecting more important sentences than model B.

4.2.2 CIQ Evaluation Result

We compute the CIQ score for the same 4798 documents evaluated in Section 4.1.2 to evaluate our generated narratives based on captured importance. We also calculate the scaled salience for the sentences in each gold narrative. Figure 4.2 shows that among the 4798 documents, Sumsage yields a higher CIQ score for 57.27% of all the documents (2748 documents) while GPT-3.5-turbo yields a higher CIQ in only 31.35% of the total test documents (1504 documents). The equal CIQ values were obtained in 11.38% of the documents (546 documents).

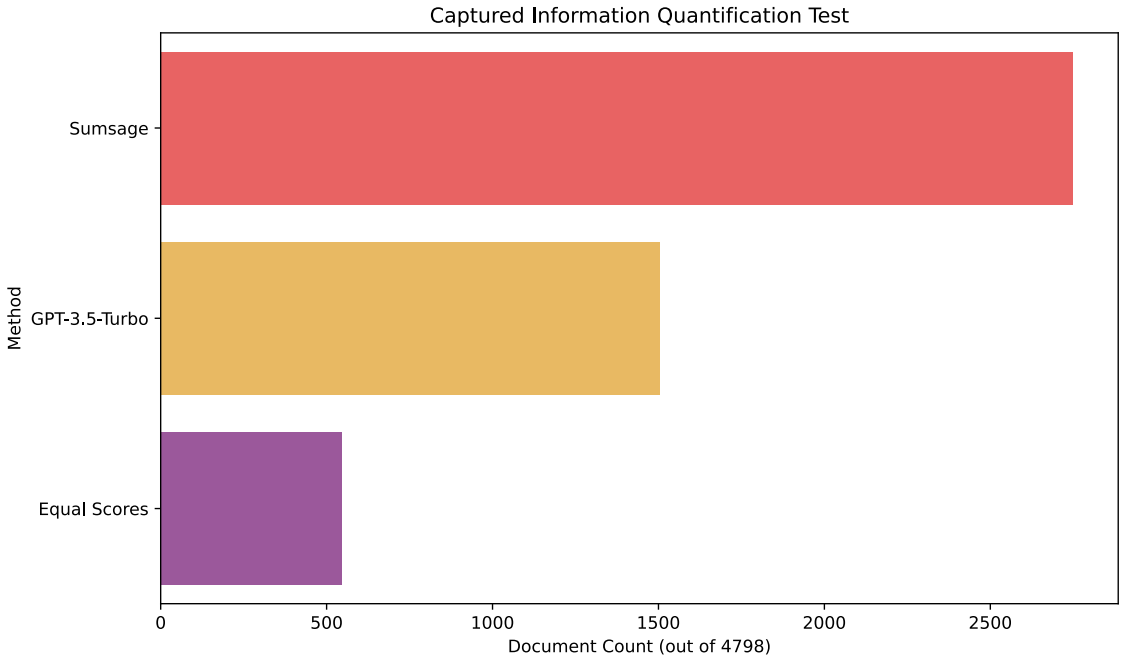


Figure 4.2: Count of the cases where a model scored higher than the other in the CIQ test.

To further evaluate the effectiveness of the sentence selection methods, we analyzed the percentage of sentences captured by both the intersection of the gold standard with the GPT-3.5-turbo generated sentences and the intersection of the gold standard with the Sumsage generated sentences, relative to the total sentences in the gold standard for each document.

These percentages were then averaged across all documents to obtain separate grand averages for the GPT-3.5-turbo and Sumsage summaries. The equation to calculate the grand average is as follows:

$$\text{Grand Average} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\text{Number of sentences in the intersection}_i}{\text{Total number of sentences in the gold standard}_i}\right) \times 100,$$

where N is the total number of documents.

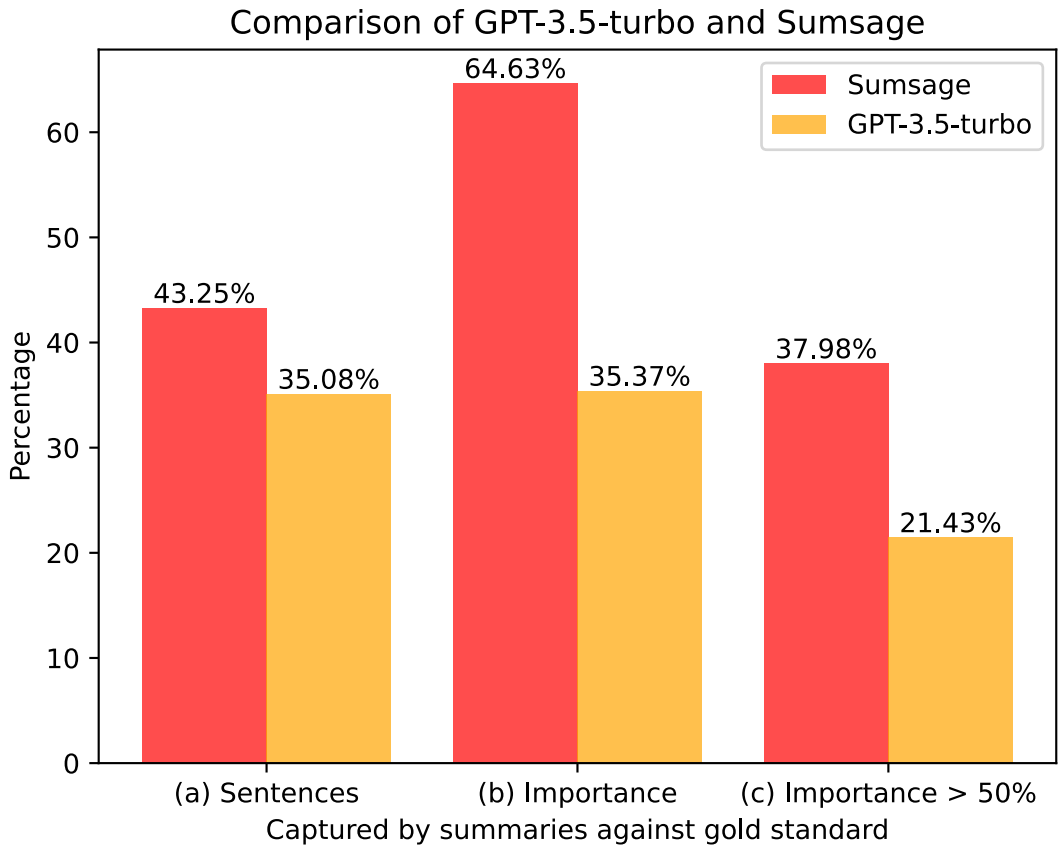


Figure 4.3: Overall percentage of captured (a) sentences, (b) importance, and (c)over 50% importance by the summarizers in comparison to the gold standard.

The results indicate that the intersection of the gold standard with the GPT-3.5-turbo

generated sentences captured an average of approximately 35.08% of the sentences, while the intersection of the gold standard with the Sumsage generated summary sentences captured an average of approximately 43.25% of the sentences (Figure 4.3(a)). These grand averages provide a comprehensive measure of the sentence selection performance, highlighting the proportion of gold standard sentences effectively identified by the GPT-3.5-turbo and Sumsage methods across the entire dataset. Moreover, according to CIQ, the sentences in the intersection capture 35.37% and 64.63% of the information presented in the gold standard narrative for summaries generated by GPT-3.5-turbo and Sumsage respectively (Figure 4.3(b)).

We conducted an additional analysis to evaluate the effectiveness of the summarizers in capturing more than 50% of the overall importance within documents. The objective was to determine the proportion of documents for which each summarizer could extract the majority of key information as determined by the gold standard. Our experiment revealed that GPT-3.5-Turbo was able to capture more than 50% of the overall importance present in the gold standard in 21.43% of the documents. In contrast, Sumsage demonstrates significantly better performance, capturing over 50% of the overall importance in 37.98% of the documents (Figure 4.3(c)).

Considering Example 1 from the CIQ perspective reveals that Sumsage covers all the key information presented by the human annotator. Consequently, the Sumsage method scores 100 on the CIQ scoring scale. However, GPT-3.5-Turbo missed the most important sentence: "under-fire France boss **Philippe Saint-André** has rung the changes following his side's **20-13 loss** to **Wales** in Paris last time out." which contains four critical pieces of information on its own. As a result, GPT-3.5-Turbo is scored down to 50.60 in the CIQ scoring method.

Analyzing Example 2 from the CIQ perspective shows that while Sumsage misses some important information, GPT-3.5-Turbo covered all the key information identified by the

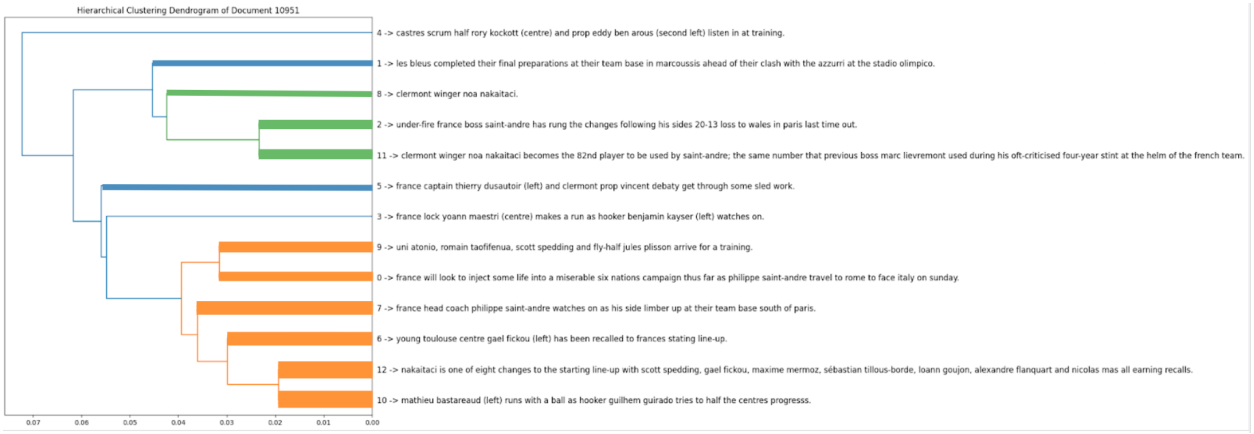
human annotator. The phrase "passed away" in the summary generated by GPT-3.5-turbo was considered equivalent to the word "die" in the gold standard summary. As a result, the summary generated by GPT-3.5-Turbo achieved a perfect CIQ score of 100, while Sumsage scored 54.03.

CHAPTER 5
DISCUSSION

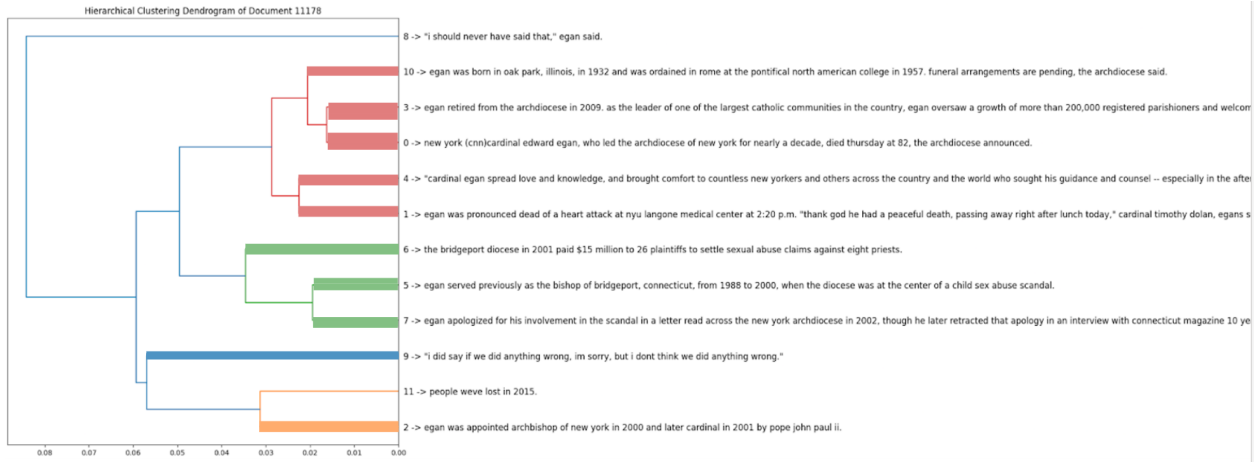
5.1 Evaluation

Sumsage is an optimization-based text summarization method designed to emulate the approach of an expert human writer in selecting important sentences. When summarizing a document, a writer typically does not include all topics in a document during summarization. Instead, they focus on a few key sentences. Our analysis of the CNN/DM dataset reveals that human annotators typically use information from only 4% of all available sentences on average when summarizing. By identifying the most salient sentences, Sumsage is able to effectively reduce the size of input documents while preserving the essential information.

Sumsage achieves effective text summarization through the use of hierarchical clustering and a penalty function. Hierarchical clustering arranges sentences based on their embeddings, forming a tree structure that encapsulates the document's thematic organization. During sentence selection, Sumsage utilizes the physical distance between the sentences in this tree to compute matching and ordering penalties. The penalty function (Figure 3.6) imposes high penalties on sentences that are either too distant (suggesting unrelated content) or too proximate (indicating redundancy) to the selected sentence. This penalization process ensures that the selected sentences do not jump between topics by avoiding the selection of the farthest sentences, which can cause frequent topic jumps, and the closest sentences which can lead to redundancy. Therefore, by utilizing the designed penalty function we can ensure that the chosen sentences are semantically related yet distinct, thereby minimize topic jumps and redundancy. By concentrating on the most salient sentences as humans, Sumsage efficiently condenses the document while preserving essential information, closely emulating the approach of expert human summarizers.



(a) Document in example 1



(b) Document in example 2

Figure 5.1: Hierarchical Clustering of the sentences in the document for (a) Example 1 and (b) Example 2. The thickness on the edges indicate the importance of the sentences. The thickness is calculated using the cosine similarity between the sentence embeddings and the document embedding.

Figure 5.1 illustrates the hierarchical clusterings of the documents where the roots are considered the document representations and the leaves represent each sentence. The clustering trees are generated with the ward method and with cosine distance set as the metric. The clusters are shown as different colors and the threshold for the cut-off point was manually set to 0.045 for these documents. As we don't use the clusters in any calculation, manual thresholding is used only for visualization purposes.

Upon running the IBA on the summaries of Example 1, we found that the sentences used by the Gold standard summary and the Sumsage algorithm were sentences 2, 11, 12, and 6, in that order. However, GPT-3.5-turbo generated the summary using sentences 0, 12, 11, and 1 in this specific order. This indicates that GPT-3.5-turbo matched only two sentences from the Gold standard narrative, and those matched sentences were in a different order. The order between the two matches (sentences 11 and 12) is significant because we consider the physical distance between each pair of sentences in the evaluation. Consequently, GPT-3.5-turbo was penalized in both the matching and ordering aspects of the evaluation, resulting in a higher Symphony penalty. In contrast, Sumsage generated the same sentences and laid them in the same order as a human-generated summary, resulting in a Symphony penalty of 0.0. As illustrated in Figure 5.1(a), d(2,11) + d(11,12) + d(12,6) is much less than d(0,12) + d(12,11) + d(11,1), where d represents the physical distance between a pair of sentences in the tree. By minimizing the total physical distance of the generated narrative, we ensure that the summarizer produces a narrative similar to that of the human summary writers.

The CIQ measure is particularly sensitive to the omission of high-importance sentences, which are determined based on the sentences used by the humans. This sensitivity arises from the need to capture the most crucial information in a summary. The CIQ evaluates how well a summarization method captures the essential sentences, i.e., ideas, in the original document and assigns higher scores to summaries that include sentences deemed highly important by human summary writers.

For example, in the case of news articles, certain sentences may contain important information such as key events, outcomes, or quotes that are critical for understanding the overall context. The ability to identify and include these important sentences directly impacts the effectiveness of the summary. Sumsage's approach to leveraging hierarchical clustering and physical distance helps ensure that these critical sentences are selected and appropriately ordered, thereby aligning more closely with human-generated summaries.

In contrast, GPT-3.5-turbo, while powerful in generating coherent and fluent summaries, may not always align with human judgment regarding sentence importance and ordering. This misalignment can lead to the exclusion of crucial information and a higher Symphony penalty, as observed in Example 1. Therefore, while both methods have their strengths, the ability of Sumsage to more accurately mimic human summarization practices makes it particularly effective in contexts where capturing the most important information is paramount.

An alternate method to evaluate the results is to ask human annotators which summarizer they prefer. But we reject that idea for two main reasons. Firstly, it is a mundane manual task for human annotators to read thousands of news articles and their summaries. This may lead to low-quality annotation. Secondly, this does not quantify how much information the summaries are capturing nor if the summaries follow the human-written summaries. Therefore we use only the automated evaluation methods for analyzing the summaries in the thesis. According to them, the Sumsage technique demonstrates a promising approach to summarization, aligning closely with human annotators' methods. Future research could explore further refinements to enhance conciseness and extend this approach to other datasets and domains.

5.2 Sumsage against the Open Challenges in Summarization

Advancements in NLP continuously introduce novel techniques for automated text summarization. However, each breakthrough also brings forth unique challenges that require careful consideration to enable further progress in the field. For instance, even though LLMs have significantly enhanced text summarization, they continue to face challenges in preserving contextual relevance, managing computational efficiency, ensuring the interpretability of summarization models, and achieving dynamic summarization. As the field advances, addressing these issues will be essential for achieving reliable and effective text summarization. Outlined below are some of the primary challenges we believe remain difficult to address with LLMs, along with strategies employed by Sumsage to manage these issues.

5.2.1 Hallucination

LLMs, while powerful, often suffer from hallucination, where the model generates information not present in the source text. This occurs due to the extensive pre-training on diverse datasets, which sometimes leads the models to draw on information outside the given context. Sumsage addresses this by following an extract-then-abstract approach. Focusing on extractive summarization first ensures that only relevant and important sentences are considered before applying abstraction. This method reduces the context size for LLMs from the entire document to just the selected sentences, significantly minimizing the risk of hallucination. As a result, Sumsage preserves the contextual relevance of the generated summaries by closely aligning them with the source document's content.

5.2.2 Computational Efficiency

As the complexity and size of LLMs increase, so do the computational requirements. Handling large documents with extensive context windows can be computationally intensive and time-consuming. Sumsage mitigates this by leveraging tree data structures and hierarchical clustering which allows for efficient data navigation and retrieval, making the summarization process scalable and theoretically unbounded. The computational complexity of Sumsage is $O(\mid D \mid \cdot n \cdot \mid g \mid)$, where $\mid D \mid$ is the number of documents, n is the number of sentences per document, and $\mid g \mid$ is the average size of the gold standard narrative. Hierarchical clustering organizes sentences based on their embeddings, creating a tree that can be easily traversed to extract relevant information. This method not only enhances computational efficiency but also enables the model to handle larger documents effectively without a significant increase in processing time.

5.2.3 Interpretability

Interpretability is a significant concern with many deep learning models, including LLMs. Understanding why a model selects certain sentences over others is crucial for trust and usability. Sumsage enhances interpretability through its mathematical penalty function used for sentence selection. This function assigns importance scores to sentences based on their cosine similarity with the document embedding and applies a penalty to sentences that are too similar or too distant from the already selected ones. This transparent scoring mechanism makes it clear why certain sentences are included in the summary, providing a level of interpretability often lacking in other models. By visualizing the hierarchical clustering tree and the associated penalties, users can understand and trust the summarization process.

5.3 Limitations and Future Work

While the optimized mathematical penalty function for the CNN/DM dataset performs well on test documents, the limited scope of our experiments, which focused solely on news articles, prevents us from concluding whether this optimized penalty function is equally effective for other datasets or types of documents, such as research papers. In the future, we intend to explore this question with research publications and summaries obtained from abstracts. We plan to explore reinforcement learning (RL), which we believe will enable us to fine-tune the model's parameters in real time, further improving its performance across various domains. Through these efforts, we hope to develop a more versatile and adaptive summarization method capable of handling a broader array of document types effectively.

The ability to dynamically adjust summaries, i.e. using a human-in-the-loop-based approach is another challenge in text summarization. Traditional methods often produce static summaries, which do not change once generated. Sumsage, however, can leverage structures to support dynamic summarization. Users could prune branches of the tree, effectively excluding certain topics or details from the summary. The model would regenerate the summary based on the updated tree structure, providing an adaptable summarization tool that can cater to different user preferences and contexts. This dynamic capability is particularly useful in applications requiring real-time updates or customization based on specific criteria.

The proposed symphony scoring mechanism is more effective and intuitive than the ROUGE metrics for evaluating summaries generated by Sumsage. It enables the scoring system to consider whether the summary selects the same sentences as the human expert in the same order. However, the symphony penalty can yield good results even if the generated narrative misses the most important sentences in comparison to the gold standard but matches many less important sentences. For example, if a human writer used a set of sentences, S = {1,4,6,12,18} in the summary with the quantified importance of each sentence

being respectively 70%, 10%, 5%, 5% and 10%, totaling 100%, a summarizer could match all of the sentences while missing the most important sentence (sentence number 1) and still get a very low symphony. We check this behavior using another process that we call the CIQ technique which measures how much of the importance the generated summary is capturing. In the future, we plan to integrate the CIQ scoring mechanism into the Symphony algorithm so that the limitation is resolved and the Symphony algorithm sufficiently evaluates the performance of the generated narrative.

Another limitation of the Symphony evaluation penalty is its inability to assess the conciseness of summaries. A summarizer could easily deceive the Symphony penalty by producing an excessively long summary or even by presenting the entire input document as the summary, thus gaining a perfect symphony and CIQ score. Future research will further focus on incorporating conciseness evaluation into Symphony, aiming to enhance its robustness as an evaluation penalty.

CHAPTER 6
CONCLUSION

In this research, we explore and address the limitations of traditional text summarization techniques and the current open challenges introduced by Large Language Models (LLMs). Our research introduces Sumsage, a novel optimization-based approach to text summarization that leverages hierarchical clustering within tree data structures to enhance narrative synthesis. Our contributions are threefold. Firstly, we develop the Syn-D-sum dataset, a synthetic summary dataset derived from the CNN/DailyMail dataset, providing a valuable resource for training and evaluating summarization models. Secondly, we propose the Sumsage algorithm, which employs hierarchical clustering to extract salient sentences and construct coherent summaries, closely mimicking the human summarizers' narrative structures. Thirdly, we introduced the Symphony evaluation penalty, designed to better assess the quality of generated summaries by considering both the narrative structure and the order of sentences.

Through extensive experiments and evaluation, we demonstrate that Sumsage outperforms GPT-3.5-turbo in terms of generating human-like text summaries. Our results indicate that Sumsage captures more information and aligns more closely with human-generated summaries, as evidenced by the Symphony score and the Captured Importance Quantification (CIQ) test. Our research further contributes to addressing several critical challenges in text summarization. By focusing on the extractive summarization component and leveraging LLMs for abstraction, we mitigate issues like hallucination and preserve contextual relevance. The dynamic nature of the tree structure and the hierarchical clustering approach enables efficient and scalable summarization, while the interpretability of the mathematical penalty function enhances the transparency of the overall summarization process.

In conclusion, Docusage represents a significant advancement in the field of text summa-

rization, offering a robust and interpretable method for generating high-quality summaries. Our approach not only addresses the current challenges but also lays the foundation for future innovations in narrative synthesis and evaluation.

REFERENCE

[1] Griffin Adams, Alexander R Fabbri, Faisal Ladhak, Eric Lehman, and Noémie Elhadad. From sparse to dense: Gpt-4 summarization with chain of density prompting. In *Proceedings of EMNLP Workshop*, page 68, 2023.

[2] Mousumi Akter, Naman Bansal, and Shubhra Kanti Karmaker. Revisiting automatic evaluation of extractive summarization task: Can we do better than rouge? In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1547–1560, 2022.

[3] Zakariae Alami Merrouni, Bouchra Frikh, and Brahim Ouhbi. Exabsum: a new text summarization approach for generating extractive and abstractive summaries. *Journal of Big Data*, 10(1):163, 2023.

[4] Wajdi Homaid Alquliti and Norjihan Binti Abdul Ghani. Convolutional neural network based for automatic text summarization. *International Journal of Advanced Computer Science and Applications*, 10(4), 2019.

[5] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report; 2023. *arXiv preprint arXiv:2305.10403*, 2023.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[7] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Jade Goldstein, Alon Lavie, Chin-Yew Lin, and Clare Voss, editors, *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[8] Lochan Basyal and Mihir Sanghvi. Text summarization using large language models: a comparative study of mpt-7b-instruct, falcon-7b-instruct, and openai chat-gpt models. *arXiv preprint arXiv:2310.10449*, 2023.

[9] Mohammed Salem Binwahlan, Naomie Salim, and Ladda Suanmali. Fuzzy swarm diversity hybrid model for text summarization. *Information processing & management*, 46(5):571–588, 2010.

[10] Nik Bear Brown. Enhancing trust in llms: Algorithms for comparing and interpreting llms. *arXiv preprint arXiv:2406.01943*, 2024.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[12] Rodrigo C Camargos, Paulo R Nietto, and Maria do Carmo Nicoletti. Agglomerative and divisive approaches to unsupervised learning in gestalt clusters. In *Intelligent Systems Design and Applications: 16th International Conference on Intelligent Systems Design and Applications (ISDA 2016) held in Porto, Portugal, December 16-18, 2016*, pages 35–44. Springer, 2017.

[13] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336, 1998.

[14] Edward Y Chang. Socrasynth: Multi-llm reasoning with conditional statistics. *arXiv preprint arXiv:2402.06634*, 2024.

[15] Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. Booookscore: A systematic exploration of book-length summarization in the era of llms. *arXiv preprint arXiv:2310.00785*, 2023.

[16] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, 2016.

[17] Sangwoo Cho, Logan Lebanoff, Hassan Foroosh, and Fei Liu. Improving the similarity measure of determinantal point processes for extractive multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1027–1038, 2019.

[18] Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760, 2019.

[19] Arman Cohan, Nazli Goharian, et al. 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)*, 2018.

[20] Michael Collins. Statistical machine translation: Ibm models 1 and 2. *Columbia Columbia Univ*, 2011.

[21] Narayana Darapaneni, Anwesh Reddy Paduri, B. G. Sudha, Adithya Kashyap, Roopak Mayya, C. S. Thejas, K. S. Nagullas, Ashwini Kulkarni, and Ullas Dani. Advanced pointer-generator networks based text generation. In *International Conference on Worldwide Computing and Its Applications*, pages 537–548, Singapore, 1997. Springer Nature Singapore.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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, June 2019. Association for Computational Linguistics.

[23] Harold P. Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[24] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[25] Kaito Fujiwara, Miyu Sasaki, Akira Nakamura, and Natsumi Watanabe. Measuring the interpretability and explainability of model decisions of five large language models. *osf*.

[26] Mahak Gambhir and Vishal Gupta. Deep learning-based extractive text summarization with word-level attention mechanism. *Multimedia Tools and Applications*, 81(15):20829–20852, 2022.

[27] Kavita Ganesan. Rouge 2.0: Updated and improved measures for evaluation of summarization tasks. *arXiv preprint arXiv:1803.01937*, 2018.

[28] Nikolaos Giarelis, Charalampos Mastrokostas, and Nikos Karacapilidis. Abstractive vs. extractive summarization: An experimental review. *Applied Sciences*, 13(13):7620, 2023.

[29] Jade Goldstein, Vibhu O. Mittal, Jaime G. Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*, 2000.

[30] Tanya Goyal, Junyi Jessy Li, and Greg Durrett. News summarization and evaluation in the era of gpt-3. *arXiv preprint arXiv:2209.12356*, 2022.

[31] Tanya Goyal, Junyi Jessy Li, and Greg Durrett. Snac: Coherence error detection for narrative summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 444–463, 2022.

[32] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[33] Puruso Muhammad Hanunggul and Suyanto Suyanto. The impact of local attention in lstm for abstractive text summarization. In *2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pages 54–57. IEEE, 2019.

[34] Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1515–1520, 2013.

[35] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[36] Kung-Hsiang Huang, Philippe Laban, Alexander R Fabbri, Prafulla Kumar Choubey, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. Embrace divergence for richer insights: A multi-document summarization benchmark and a case study on summarizing diverse information from news articles. *arXiv preprint arXiv:2309.09369*, 2023.

[37] Hanlei Jin, Yang Zhang, Dan Meng, Jun Wang, and Jinghua Tan. A comprehensive survey on process-oriented automatic text summarization with exploration of llm-based methods. *arXiv preprint arXiv:2403.02901*, 2024.

[38] Cameron R Jones and Benjamin K Bergen. People cannot distinguish gpt-4 from a human in a turing test. *arXiv preprint arXiv:2405.08007*, 2024.

[39] Jee-weon Jung, Roshan Sharma, William Chen, Bhiksha Raj, and Shinji Watanabe. Augsumm: Towards generalizable speech summarization using synthetic labels from large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12071–12075. IEEE, 2024.

[40] Rishabh Karmakar, Ketki Nirantar, Prathamesh Kurunkar, Pooja Hiremath, and Deptii Chaudhari. Indian regional language abstractive text summarization using attention-based lstm neural network. In *2021 International Conference on Intelligent Technologies (CONIT)*, pages 1–8. IEEE, 2021.

[41] Kazuya Kawakami. *Supervised sequence labelling with recurrent neural networks*. PhD thesis, University of Toronto, 2008.

[42] Gil Keren and Björn Schuller. Convolutional rnn: An enhanced model for extracting features from sequential data. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3412–3419. IEEE, 2016.

[43] Y Kim. Convolutional neural networks for sentence classification. arxiv [j]. *arXiv preprint*, 2014.

[44] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[45] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[46] Harsh Kumar, Gaurav Kumar, Shaivye Singh, and Sourav Paul. Text summarization of articles using lstm and attention-based lstm. In *Machine Learning and Autonomous*

Systems: Proceedings of ICMLAS 2021, pages 133–145. Springer Nature Singapore, Singapore, 2022.

[47] Litton J Kurisinkel and Nancy F Chen. Llm based multi-document summarization exploiting main-event biased monotone submodular content extraction. *arXiv preprint arXiv:2310.03414*, 2023.

[48] Philippe Laban, Alexander R Fabbri, Caiming Xiong, and Chien-Sheng Wu. Summary of a haystack: A challenge to long-context llms and rag systems. *arXiv preprint arXiv:2407.01370*, 2024.

[49] Salima Lamsiyah, Abdelkader El Mahdaouy, Saïd El Alaoui Ouatik, and Bernard Espinasse. Unsupervised extractive multi-document summarization method based on transfer learning from bert multi-task fine-tuning. *Journal of Information Science*, 49(1):164–182, 2023.

[50] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[51] Zekai Li, Yanxia Qin, Qian Liu, and Min-Yen Kan. Isqa: Informative factuality feedback for scientific summarization. *arXiv preprint arXiv:2404.13246*, 2024.

[52] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, 2004.

[53] Hui Lin and Jeff Bilmes. 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, 2011.

[54] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[55] Yang Liu and Mirella Lapata. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, page 5070. Association for Computational Linguistics, 2019.

[56] Yang Liu and Mirella Lapata. 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, 2019.

[57] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[58] Yixin Liu, Budhaditya Deb, Milagro Teruel, Aaron Halfaker, Dragomir Radev, and Ahmed Hassan. On improving summarization factual consistency from natural language feedback. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15144–15161, 2023.

[59] Yixin Liu and Pengfei Liu. Simcls: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, 2021.

[60] Mrs Meghana P Lokhande, Mrs Namrata Gawande, Mrs Shweta Koprade, and MM Bewoor. Text summarization using hierarchical clustering algorithm and expectation maximization clustering algorithm. *Int. J. Comput. Eng. Technol.(IJCET)*, 6:58–65, 2015.

[61] Cedric Lothritz, Kevin Allix, Lisa Veiber, Jacques Klein, and Tegawendé François D Assise Bissyande. Evaluating pretrained transformer-based models on the task of fine-grained named entity recognition. In *28th International Conference on Computational Linguistics*, 2020.

[62] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[63] Haoyan Luo and Lucia Specia. From understanding to utilization: A survey on explainability for large language models. *arXiv preprint arXiv:2401.12874*, 2024.

[64] Ayuns Luz. Enhancing the interpretability and explainability of ai-driven risk models using llm capabilities. Technical report, EasyChair, 2024.

[65] Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. Multi-document summarization via deep learning techniques: A survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[66] Inderjeet Mani and Mark T. Maybury. *Advances in Automatic Text Summarization*. MIT Press, Cambridge, MA, 1999.

[67] Abu Kaisar Mohammad Masum, Sheikh Abujar, Md Ashraful Islam Talukder, AKM Shahariar Azad Rabby, and Syed Akhter Hossain. Abstractive method of text summarization with sequence to sequence rnns. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–5. IEEE, 2019.

[68] Ryan McDonald. A study of global inference algorithms in multi-document summarization. In *European Conference on Information Retrieval*, pages 557–564, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[69] Patrick E. McKnight and Julius Najab. Mann-whitney u test. In *The Corsini Encyclopedia of Psychology*, pages 1–1. John Wiley & Sons, Inc., 2010.

[70] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[71] Muhammad Firoz Mridha, Aklima Akter Lima, Kamruddin Nur, Sujoy Chandra Das, Mahmud Hasan, and Muhammad Mohsin Kabir. A survey of automatic text summarization: Progress, process and challenges. *IEEE Access*, 9:156043–156070, 2021.

[72] Wenchuan Mu and Kwan Hui Lim. Universal evasion attacks on summarization scoring. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 104–118, 2022.

[73] Ani Nenkova, Kathleen McKeown, et al. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233, 2011.

[74] Frank Nielsen and Frank Nielsen. Hierarchical clustering. *Introduction to HPC with MPI for Data Science*, pages 195–211, 2016.

[75] Andrei Olariu. Hierarchical clustering in improving microblog stream summarization. In *Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24-30, 2013, Proceedings, Part II 14*, pages 424–435. Springer, 2013.

[76] Ji Pei, Rim Hantach, Sarra Ben Abbès, and Philippe Calvez. Towards hybrid model for automatic text summarization. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 987–993. IEEE, 2020.

[77] Marco Polignano, Pierpaolo Basile, Marco de Gemmis, and Giovanni Semeraro. A comparison of word-embeddings in emotion detection from text using bilstm, cnn and self-attention. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, pages 63–68, 2019.

[78] Xiao Pu, Mingqi Gao, and Xiaojun Wan. Summarization is (almost) dead. *arXiv e-prints*, 2023.

[79] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[80] Herbert E. Robbins. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[81] Kuniaki Saito, Kihyuk Sohn, Chen-Yu Lee, and Yoshitaka Ushiku. Unsupervised llm adaptation for question answering. *arXiv preprint arXiv:2402.12170*, 2024.

[82] Brenda Santana, Ricardo Campos, Evelin Amorim, Alípio Jorge, Purificação Silvano, and Sérgio Nunes. A survey on narrative extraction from textual data. *Artificial Intelligence Review*, 56(8):8393–8435, 2023.

[83] Prachi Shah and Nikita P Desai. A survey of automatic text summarization techniques for indian and foreign languages. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pages 4598–4601. IEEE, 2016.

[84] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 28, 2015.

[85] Chandan Singh, Jeevana Priya Inala, Michel Galley, Rich Caruana, and Jianfeng Gao. Rethinking interpretability in the era of large language models. *arXiv preprint arXiv:2402.01761*, 2024.

[86] Shengli Song, Haitao Huang, and Tongxiao Ruan. Abstractive text summarization using lstm-cnn based deep learning. *Multimedia Tools and Applications*, 78(1):857–875, 2019.

[87] Melanie Subbiah, Faisal Ladhak, Akankshya Mishra, Griffin Adams, Lydia B Chilton, and Kathleen McKeown. Storysumm: Evaluating faithfulness in story summarization. *arXiv preprint arXiv:2407.06501*, 2024.

[88] Shichao Sun, Ruifeng Yuan, Ziqiang Cao, Wenjie Li, and Pengfei Liu. Prompt chaining or stepwise prompt? refinement in text summarization. *arXiv preprint arXiv:2406.00507*, 2024.

[89] Hiroya Takamura and Manabu Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 781–789, 2009.

[90] Liyan Tang, Igor Shalyminov, Amy Wong, Jon Burnsky, Jake Vincent, Siffi Singh, Song Feng, Hwanjun Song, Hang Su, Lijia Sun, et al. Tofueval: Evaluating hallucinations of llms on topic-focused dialogue summarization. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4455–4480, 2024.

[91] Oguzhan Tas and Farzad Kiyani. A survey automatic text summarization. *PressAcademia Procedia*, 5(1):205–213, 2007.

[92] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[93] Danny CL Tsai, W Chang, and S Yang. Short answer questions generation by fine-tuning bert and gpt-2. In *Proceedings of the 29th International Conference on Computers in Education Conference, ICCE*, volume 64, 2021.

[94] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[95] Pradeepika Verma and Anshul Verma. A review on text summarization techniques. *Journal of scientific research*, 64(1):251–257, 2020.

[96] Seema Wazarkar, Bettahally N Keshavamurthy, and Amrita Manjrekar. A review of hierarchical fuzzy text clustering. *ADVANCED COMPUTING (ICoAC 2017)*, page 516, 2017.

[97] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[98] Yuxiang Wu and Baotian Hu. Learning to extract coherent summary via deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[99] Liqiang Xiao, Lu Wang, Hao He, and Yaohui Jin. Copy or rewrite: Hybrid summarization with hierarchical reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9306–9313, 2020.

[100] Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. C-pack: Packaged resources to advance general chinese embedding, 2023.

[101] Wen Xiao and Giuseppe Carenini. Extractive summarization of long documents by combining global and local context. *arXiv preprint arXiv:1909.08089*, 2019.

[102] Huiyan Xu, Zhijian Wang, and Xiaolan Weng. Scientific literature summarization using document structure and hierarchical attention model. *IEEE Access*, 7:185290–185300, 2019.

[103] Chengran Yang, Jiakun Liu, Bowen Xu, Christoph Treude, Yunbo Lyu, Ming Li, and David Lo. Apidocbooster: An extract-then-abstract framework leveraging large language models for augmenting api documentation. *arXiv preprint arXiv:2312.10934*, 2023.

[104] Jia-Yu Yao, Kun-Peng Ning, Zhen-Hui Liu, Mu-Nan Ning, and Li Yuan. Llm lies: Hallucinations are not bugs, but features as adversarial examples. *arXiv preprint arXiv:2310.01469*, 2023.

[105] ChengXiang Zhai. Probabilistic topic models for text data retrieval and analysis. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 1399–1401, 2017.

[106] Haopeng Zhang, Xiao Liu, and Jiawei Zhang. Extractive summarization via chatgpt for faithful summary generation. *arXiv preprint arXiv:2304.04193*, 2023.

[107] Haopeng Zhang, Xiao Liu, and Jiawei Zhang. Summit: Iterative text summarization via chatgpt. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10644–10657, 2023.

[108] Haopeng Zhang, Philip S Yu, and Jiawei Zhang. A systematic survey of text summarization: From statistical methods to large language models. *arXiv preprint arXiv:2406.11289*, 2024.

[109] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[110] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[111] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 2024.

[112] Yusen Zhang, Nan Zhang, Yixin Liu, Alexander Richard Fabbri, Junru Liu, Ryo Kamoi, Xiaoxin Lu, Caiming Xiong, Jieyu Zhao, Dragomir Radev, et al. Fair abstractive summarization of diverse perspectives. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3404–3426, 2024.

[113] Jintao Zhao, Libin Yang, and Xiaoyan Cai. Hettreesum: A heterogeneous tree structure-based extractive summarization model for scientific papers. *Expert Systems with Applications*, 210:118335, 2022.

[114] Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. 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 563–578, 2019.

[115] Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. Towards a unified multi-dimensional evaluator for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2023–2038, 2022.

[116] Irune Zubiaga, Aitor Soroa, and Rodrigo Agerri. A llm-based ranking method for the evaluation of automatic counter-narrative generation. *arXiv preprint arXiv:2406.15227*, 2024.