GRAPHSUMM: Graphical Text Summarization Using Generative AI

Jong Inn Park[◊]
[◊]University of Minnesota
park2838@umn.edu

Abstract

We propose an innovative end-to-end approach, GRAPHSUMM, to summarize and visualize transcribed text data from speeches, such as meeting notes, which are often unstructured and multidimensional. Leveraging advancements in Automatic Speech Recognition (ASR) and Generative AI, this work aims to transform long, text-based summaries into structured, graphical visualizations, thus enhancing accessibility and comprehension. Traditional text summaries, while organized, fail to offer an immediate understanding of the key points and topic structure of speeches. Our method employs ASR technology, notably OpenAI's Whisper (Radford et al., 2022), to transcribe spoken content into text, which is then processed using various summarization modes customized to the content's nature—such as timelines and topic clustering. These summaries are enriched with additional information and structured to highlight significant content, intending to facilitate a deeper and quicker comprehension through graphical representation. This approach aims to bridge the gap in current speech summarization tools by providing a visual summary that can significantly improve user engagement and understanding, especially in contexts like meetings or Q&A sessions where multiple topics and speakers are involved.[1]

1 Introduction

The proliferation of digital communication has led to an increasing need for effective summarization (Cao et al., 2018) and visualization tools (Vig et al., 2021) for transcribed speech data. Traditional text summaries often lack the immediate accessibility and comprehension needed for quick decision-making. This work addresses these challenges by proposing an innovative end-to-end approach that leverages Automatic Speech Recognition (ASR) and Generative AI to transform transcribed speech into structured graphical visualizations. This method aims to enhance user engagement and understanding, particularly in complex settings such as meetings or Q&A sessions where multiple topics and speakers are involved.

[1]The code is available at https://github.com/jong-inn/GraphicalSumm.

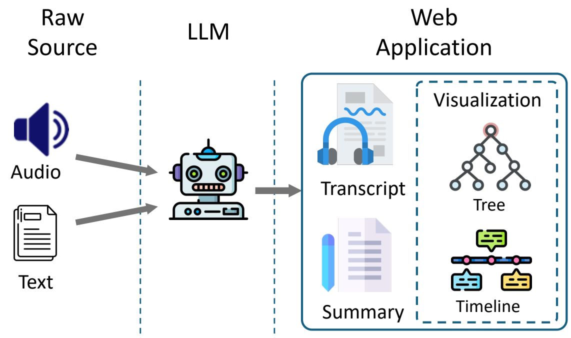

Figure 1: General process of current web applications that display transcripts and summaries from long audio or text resources and GRAPHSUMM's visualization contribution.

While transcription and summarization technologies have advanced significantly, they primarily focus on generating text-based outputs, which may not effectively convey the hierarchical and contextual relationships inherent in speech data. Existing solutions often fall short in providing an immediate understanding of key points and the overall structure of the content. This project builds on these technologies, introducing graphical summaries to address these limitations (Figure 1).

As the demand for unstructured data processing grows, leveraging ASR and Generative AI offers a promising solution. This project utilizes OpenAI's Whisper (Radford et al., 2022) for accurate transcription and GPT-4 (OpenAI et al., 2024) for creating both extractive and abstractive summaries. These summaries are converted into graphical formats to facilitate a deeper and quicker comprehension of the content. The architecture of our system integrates these components seamlessly, ensuring a robust workflow from audio input to graphical output. We summarize our contributions as follows:

- We introduce GRAPHSUMM, the first structured summary visualization tool.
- The tool enhances users' understanding of a long speech or text material.

2 Background and Related Work

2.1 Speech Recognition

Automatic Speech Recognition (ASR) technology has made significant strides over the past few decades, transforming how we convert spoken language into text. ASR employs advanced algorithms and machine learning models to handle variations in speech patterns, accents, and noise interference. Traditionally, ASR relied on large transcribed speech datasets for training. However, the advent of Wav2Vec 2.0 (Baevski et al., 2020) introduced a self-supervised learning method that reduced the dependency on labeled data, marking a substantial leap in speech recognition capabilities. Despite this progress, Wav2Vec 2.0 still required a fine-tuning stage, necessitating expertise in speech recognition for optimal performance. To address these challenges, researchers have explored scaling up with weakly supervised pre-training methods. One notable advancement in this area is OpenAI's Whisper model (Radford et al., 2022). Whisper leverages 680,000 hours of labeled audio data to enhance the model's robustness and generalization abilities, significantly improving transcription quality. This approach is integrated into GRAPHSUMM, which ensures high-quality transcriptions as the foundation for generating visual summaries.

2.2 Long Text Summarization

Summarization of long texts can be approached in two primary ways: abstractive and extractive summarization (Dong et al., 2018).

Abstractive Summarization: This method involves generating new sentences that convey the main ideas of the original text, often using advanced Natural Language Processing (NLP) techniques and generative models (Le and Le, 2013). Abstractive summarization aims to provide a concise and coherent summary by interpreting and rephrasing the content, rather than merely extracting portions of the text.

Extractive Summarization: In contrast, extractive summarization selects and compiles key sentences or phrases directly from the original text (Luhn, 1958). This method relies on identifying

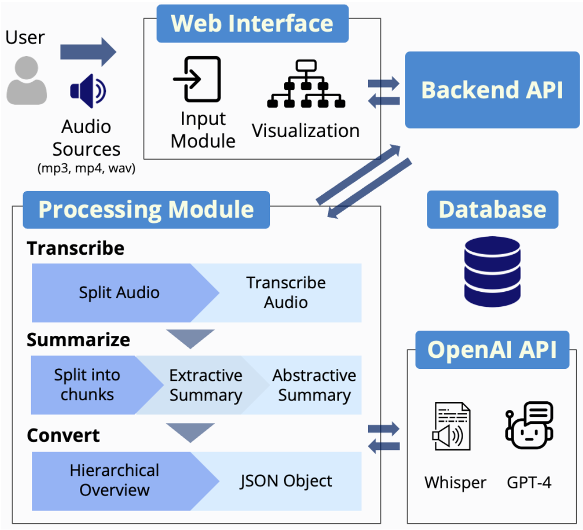

Figure 2: The architecture of GRAPHSUMM.

important segments of the text that represent the main points, without rephrasing or interpreting the information.

Both summarization techniques are employed in GRAPHSUMM to process transcribed speech data, ensuring that the generated summaries are both informative and easy to understand.

2.3 Summary Visualization

In various industries, numerous commercial products have emerged that summarize text sources and provide visualizations. In the finance sector, platforms such as (VerityPlatform, 2024) and (FactSet, 2024) released products that leverage the summarization of corporate earnings calls to generate visualized reports. These tools provide detailed visualizations that enhance the understanding of financial data.

In the domain of meeting note solutions, startups like (FireFlies, 2016) have taken a leading role. They offer a range of features for meeting recordings, including the generation of transcripts from recordings. The solution extracts keywords and tags them to the transcript, allowing users to track the topics discussed. Additionally, it shows corresponding sentences during playback, and multiple collaborators can take notes and create threads on specific parts of the transcript.

In academia, efforts have been made to display transcribed texts and analyze the outputs of language models. For instance, SummVis (Vig et al., 2021) aimed to provide users with tools to debug hallucinations in generated text using Large Language Models (LLMs).

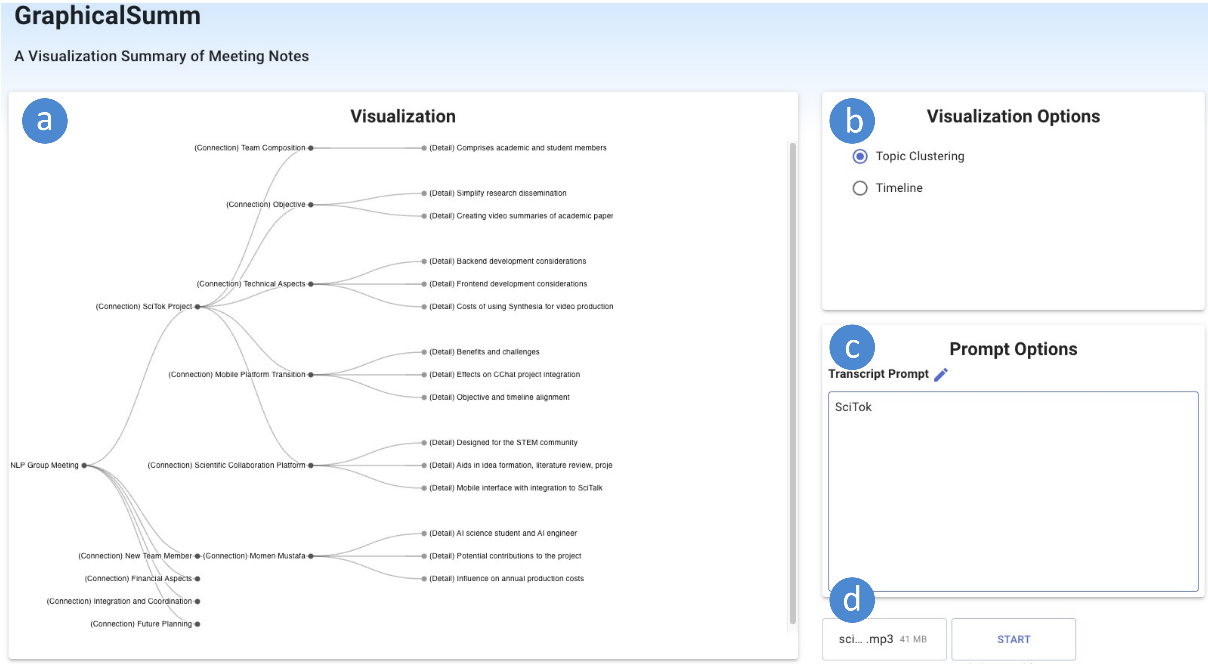

Figure 3: GRAPHSUMM Interface Components. (a) Users interact with a visualization panel by clicking visual components. (b) Users choose visualization modes. (c) Users attach a prompt to adjust words such as pronouns in transcription. (d) Buttons for uploading a file and starting the process.

Despite the practical utility of these solutions, they are primarily text-based and do not fully leverage the potential of graphical representations. GRAPHSUMM addresses this gap by utilizing D3.js (Bostock et al., 2011), a JavaScript library that binds data to the Document Object Model (DOM) and creates dynamic, interactive visual representations. By providing a graphical overview of summaries, GRAPHSUMM helps users quickly grasp the key points and structure of the content, offering an enhanced user experience.

3 GRAPHSUMM

In this section, we delve into GRAPHSUMM, our innovative system designed to transform transcribed speech data into visually structured summaries. GRAPHSUMM consumes an audio file from the users' end, shows statuses during the backend process, and finally outputs the visualization. This section provides a comprehensive overview of the system's components, their interactions, and the backend processes that enable the generation of these visual summaries.

3.1 System Architecture

The architecture of GRAPHSUMM is designed to efficiently process and visualize transcribed speech data. Figure 2 provides an overview of the system's components and their interactions.

The system comprises several key modules:

- **Web Interface:** The user interacts with the system through a web interface, which allows for audio input and visualization of the summarized data.
- **Backend API:** This module handles the communication between the web interface and the processing module.
- **Processing Module:** This module is responsible for transcribing, summarizing, and converting the audio data into a visual format. It interacts with the OpenAI API for transcription and summarization tasks.
- **Database:** Used for storing audio files, transcripts, and summaries.
- **OpenAI API:** Utilized for ASR and summarization using Whisper (Radford et al., 2022) and GPT-4 (OpenAI et al., 2024).

3.2 Backend Process

Figure 2 processing module part illustrates the detailed process flow of the backend system, from receiving audio input to generating the final visual summary. The backend process is divided into several stages:

- **Transcribe:** The audio files are first split into smaller chunks if they exceed 25MB. These chunks are then transcribed using OpenAI's Whisper (Radford et al., 2022), resulting in text files and JSON objects containing the transcriptions.
- **Summarize:** The transcribed text is split into manageable chunks. Extractive summarization is performed to identify and retain the most important sentences. Abstractive summarization then condenses these chunks into concise summaries, ensuring each group is summarized in less than five sentences.
- **Convert:** The summaries are converted into a hierarchical overview and JSON objects, which serve as the foundation for the visualizations.

The system ensures efficient processing and visualization by leveraging advanced ASR and summarization technologies, providing users with a comprehensive and easily understandable summary of the original speech content.

3.3 Prompting Technique

The effectiveness of GRAPHSUMM heavily depends on the precision and efficiency of its prompting techniques during both the summarization and conversion stages. The prompts guide the LLM to generate accurate and contextually relevant outputs. Prompts for summarization and Timeline mode are presented in Appendix A. Below, we highlight the different types of prompts in Topic Clustering mode utilized during the conversion.

- **Hierarchical Tree Prompt:** In Figure 4, this prompt guides the LLM to create an overview structure representing the main topics, subtopics, and details from the provided text material. This structure visually organizes the information, making it easier to comprehend complex relationships and hierarchies. This step helps LLM organize the summary structure properly.
- **JSON Tree Prompt:** In Figure 5, this prompt instructs the LLM to convert a hierarchical tree structure into a nested JSON format, capturing the relationships and hierarchical depth of the topics. This will be fed to visualization module to show the structured summary.

````
Your task is to construct a hierarchical tree using the
provided text material. This material includes a main
topic, subtopics, and summaries.

Identify the main topics and subtopics from the text
material. These will serve as the root nodes, branches,
and leaves of your tree, which are represented as
"[Topic]". Connect these nodes with edges, which repsent
as "(Connection)", to show the relationships between
topics. You need to replace "[Topic]" and "(Connection)"
with extracted topics and edges. The final nodes should
contain the most specific information, which could be
short sentences or keywords from the text. The depth of
your tree (how many levels of topics and subtopics) will
vary based on the complexity and breadth of the provided
text material. Follow the specified answer format without
adding explanations or deviating from the structure. Use
indentation to denote hierarchy levels within the tree.

### Answer Format
```
[Root Topic]
  [Subtopic 1]
    (Connection) [Sub-subtopic 1.1]
      (Detail) [Specific Detail 1.1.1]
      (Detail) [Specific Detail 1.1.2]
      (Detail) [Specific Detail 1.1.3]
    (Connection) [Sub-subtopic 1.2]
      (Detail) [Specific Detail 1.2.1]
  [Subtopic 2]
    (Connection) [Sub-subtopic 2.1]
    (Connection) [Sub-subtopic 2.2]
      (Detail) [Specific Detail 2.2.1]
```

### Text Material
{{TEXT}}
````

Figure 4: The hierarchical tree prompting.

````
You need to create a JSON object based on a given tree
structure. The tree represents topics and their
relationships. Each topic (node) and relationship (edge)
is shown in the tree. Convert this tree into a nested JSON
format, where each topic becomes an object, relationships
are properties, and subtopics are nested as children.
Include the relationship (edge) as a property of the child
object. If it does not, fill it out with null. Assign a
depth level starting from 0 at the root and increasing by
1 at each level down. If a topic has no subtopics (leaf
node), don't include the "children" attribute. Make sure
to maintain the JSON structure's indentation for
readability. Explanations are not allowed.

### Answer Format
```json
data = {
  "name": topic1,
  "topicDepth": 0,
  "edge": edge1,
  "children": [
    {
      "name": topic1.1,
      "topicDepth": 1,
      "edge": edge1.1,
      "children": [...]
    },
    ...
  ]
}```

### Tree Structure
{{TREE}}
````

Figure 5: The json tree prompting.

3.4 Visualization

The visualization component of GRAPHSUMM is crucial for transforming summarized textual data into comprehensible graphical representations. The system offers two primary visualization modes: Topic Clustering and Timeline. Examples are in Section 5. These visualizations enhance users' un-
````

Earnings Calls	Real Recordings
50	5

Table 1: The number of audio data used to build and test out pipelines.

derstanding of complex speech data by highlighting key topics and their relationships using d3.js (Bostock et al., 2011) JavaScript library.

Topic Clustering: Topic Clustering mode converts an JSON object into a clickable organized tree structure and provides a display to users. This mode uses nested topics as nodes, representing their relationships in front of topic names with round brackets, although these relationships are not categorized. The selection of words is based on the capabilities of the LLM. Nodes can have child nodes, which users can expand or collapse by clicking on dots.

Timeline: The Timeline mode visualizes the summarized text data chronologically. This mode is especially useful for understanding the sequence of events or topics discussed over time. The implementation of this mode also utilizes d3.js (Bostock et al., 2011) to create interactive and dynamic timelines that users can explore.

4 Data Collection

In this section, we describe the process of collecting audio data to build and test the GraphSumm pipeline. First, we planned to gather public audio data that is easily accessible and suitable for evaluating our system. Corporate earnings calls are considered as an appropriate dataset due to their multiple speakers involved conversation, typically featuring Q&A sessions with analysts. These characteristics make earnings calls a robust benchmark for real-world applications and are frequently used in finance and language model research (Mukherjee et al., 2022).

We sourced our audio dataset from the distil-whisper/earnings22 (Rio et al., 2022) repository on Huggingface, which provides complete audio files and corresponding transcriptions designed for evaluating ASR systems. From a total of 125 full audio files, we randomly select 50 files to identify potential errors during development.

Additionally, we test our system with more familiar audio recordings from research meetings in our lab. We collected five recordings, ranging from 40 minutes to an hour and a half. Two recordings were captured with a single device during lab meetings, featuring multiple speakers with varying voice qualities. Due to their private nature, we can only present partial results or visualizations from these recordings.

5 Analysis - Case Study

This section presents a real-world application of GraphSumm by analyzing a research meeting from the SciTok project, an ongoing initiative in our lab. The meeting, which lasted 40 minutes and involved six participants, was recorded via Zoom. Despite the clear audio quality, some voices overlapped, and speakers referred to a shared screen, though no video recording was available. The meeting focused on introducing the project to new members and discussing specific agendas. This case study illustrates both the strengths and limitations of GraphSumm, providing insights into its capabilities in practical scenarios.

5.1 Capturing Core Agenda

One of GraphSumm's primary objectives is to accurately capture and visualize the core agenda of lengthy and complex speech data. In the SciTok project meeting, we evaluated the system's ability to identify and highlight key points.

In the Topic Clustering mode, as shown in 6, the main topics in the first depth of the clustered topic tree cover the broad range of the 40-minute meeting. Under the "App Development" topic, there are four sub-topics with relation keywords defined at the front, and these sub-topics have further child nodes, forming a structured tree. This demonstrates GraphSumm's ability to accurately capture and organize relevant topics. In the Timeline mode, as depicted in 7, GraphSumm highlights core topics in several sentences within specific time periods. If a discussion on a certain topic continues, GraphSumm extends the time coverage accordingly.

The visualization successfully conveyed the main topics discussed, presenting them in a clear and organized manner. This capability allows users to navigate the content easily, which is especially valuable in settings requiring quick comprehension of the core agenda, such as corporate meetings or academic conferences. This approach significantly improves over traditional text-based summaries, which often fail to convey the hierarchical and con-

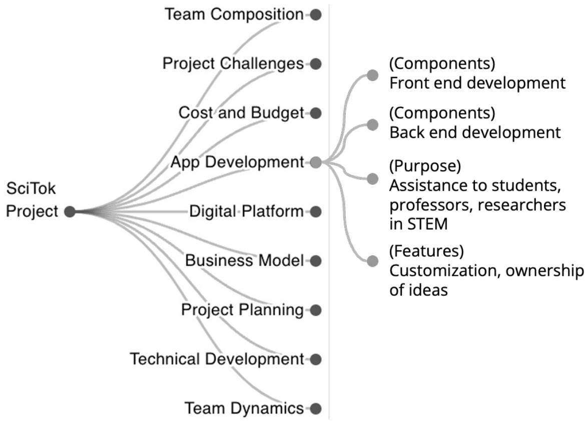

Figure 6: A simplified example of Topic Clustering mode showcasing how complex speech data is transformed into a hierarchical visualization. Each node represents a topic and can have child nodes that further detail subtopics. The leaf nodes at the ends of the branches provide final descriptions.

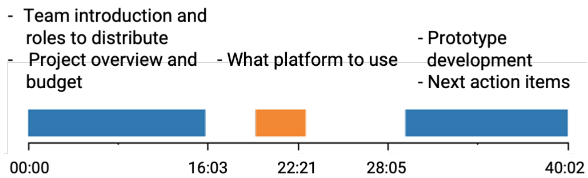

Figure 7: A simplified example of Timeline mode, highlighting how the chronological sequence of events is visualized. The bars above the timeline indicate the duration coverage of each topic discussed in the meeting. Itemized sentences describe the content within those segments.

textual relationships within the data.

5.2 Pronouns Transcription

Despite its strengths, GRAPHSUMM encountered challenges with pronoun transcription, particularly with names of people or projects. The inherent randomness of LLMs during transcription and prompting can lead to ambiguities. The accuracy of pronoun recognition is crucial for maintaining the coherence and context of the summarized content. During our evaluation, we observe instances where the system struggles with pronoun ambiguity, resulting in confusion in the final summaries.

To mitigate this issue, we incorporate additional keywords during the transcription phase to provide context for pronouns. This adjustment aims to enhance the LLM's ability to accurately interpret and transcribe pronouns. While this approach shows some improvement, it requires users to input keywords before transcription, necessitating a re-run of the system if keywords are omitted initially.

5.3 Formatting via Prompting

Another challenge encountered was related to the formatting of outputs via prompting techniques. GRAPHSUMM relies heavily on prompting to convert long texts into structured formats like JSON objects. However, inconsistencies in the responses generated by the LLM sometimes resulted in incomplete or deviated formats, affecting the final visualizations.

For example, while converting hierarchical structures into JSON in 5, deviations from the expected format sometimes lead to errors in the visualization module. These issues highlight the importance of robust and precise prompting techniques to ensure accurate and consistent outputs for effective visualizations.

6 Conclusion and Future Work

In conclusion, GRAPHSUMM demonstrates significant capabilities in capturing and visualizing complex speech data, providing a novel approach to summarization through graphical representation. The integration of Automatic Speech Recognition (ASR) with advanced generative AI techniques allows for the efficient transformation of lengthy and unstructured speech into concise and easily digestible visual summaries. This system offers enhanced user engagement and understanding, particularly in environments like corporate meetings or academic conferences where quick comprehension is crucial.

However, challenges remain in several areas. Pronoun transcription, especially concerning the names of people or projects, can sometimes lead to ambiguities, affecting the coherence of the summaries. The reliance on prompting techniques for formatting outputs into structured formats also poses issues, as inconsistencies in the generated responses can lead to errors in visualizations. To address these challenges and enhance the evaluation of GRAPHSUMM, the following future work is proposed:

- **Automated and Enhanced Evaluation:** Develop standardized evaluation metrics to objectively measure the quality of the summaries and visualizations generated by GRAPHSUMM. For evaluating summary quality, popular metrics such as ROUGE scores (Lin, 2004) can be used. Visualization evaluation could include user studies to assess the comprehension and usability compared to traditional text-based summaries.

- **Improved Pronoun Recognition:** Enhance the system's ability to accurately transcribe pronouns and context-specific terms. This could involve training custom models or incorporating additional context-aware algorithms to better handle pronoun ambiguities.
- **Various Visualization Mode:** Add more modes to convert summaries into various visualizations and display them simultaneously, providing different perspectives on the summaries. This will enhance users' understanding.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D³ data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. BanditSum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748, Brussels, Belgium. Association for Computational Linguistics.

FactSet. 2024. Factset releases transcript assistant, a game-changing ai tool for earnings call analysis. https://investor.factset.com/news-releases/news-release-details/factset-releases-transcript-assistant-game-changing-ai-tool.

FireFlies. 2016. Fireflies. https://fireflies.ai/.

Huong Thanh Le and Tien Manh Le. 2013. An approach to abstractive text summarization. In *2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR)*, pages 371–376.

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

H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

Rajdeep Mukherjee, Abhinav Bohra, Akash Banerjee, Soumya Sharma, Manjunath Hegde, Afreen Shaikh, Shivani Shrivastava, Koustuv Dasgupta, Niloy Ganguly, Saptarshi Ghosh, and Pawan Goyal. 2022. ECTSum: A new benchmark dataset for bullet point summarization of long earnings call transcripts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10893–10906, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela

Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision.

Miguel Del Rio, Peter Ha, Quinten McNamara, Corey Miller, and Shipra Chandra. 2022. "earnings-22: A practical benchmark for accents in the wild".

VerityPlatform. 2024. Earnings call insights: How investors use veritydata's new ai-powered reports. `https://verityplatform.com/resources/ai-expert-summaries-announcement/`.

Jesse Vig, Wojciech Kryscinski, Karan Goel, and Nazneen Rajani. 2021. SummVis: Interactive visual analysis of models, data, and evaluation for text summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 150–158, Online. Association for Computational Linguistics.

```
As a helpful assistant, your objective is to distill the
essence of the provided text. This involves summarizing
the content, pinpointing the main themes, and identifying
the crucial keywords. The summary should be constructed
using exact sentences from the text, with modifications
made only to clarify pronouns (e.g., changing "It shows"
to "The study shows"). Explanations are not permitted.

### Answer format
Topic: {{TOPIC}}
Keywords: {{KEYWORD1}}, {{KEYWORD2}}, ...
Summarization: {{SUMMARIZATION}}

### Text
{{TEXT}}
```

Figure 8: The extractive summary prompt.

```
You are a helpful assistant programmed to generate concise
abstractive summaries. You should capture its main themes
and insights in a condensed summary of no more than {{N}}
sentences without giving explanations.

### Text
{{TEXT}}

### Summary
```

Figure 9: The abstractive summary prompt.

```
You are tasked with creating a timeline visualization that
highlights the sequence of events or topics discussed
during a meeting. The visualization should indicate the
duration coverage of each topic and provide itemized
sentences describing the content within those segments.

Steps to generate the timeline:
1. Identify the main topics discussed during the meeting.
2. Determine the time periods during which each topic was
discussed.
3. Generate a timeline with bars indicating the duration
of each topic.
4. Provide brief descriptions of the content discussed
within each time segment.

### Format
Topic 1: [Start Time] - [End Time]
  - Brief description of what was discussed.
Topic 2: [Start Time] - [End Time]
  - Brief description of what was discussed.
...

### Example Data
Introduction: 0:00 - 5:00
  - The meeting started with an introduction to the new
project members.
Project Overview: 5:00 - 15:00
  - Overview of the project goals and timeline.
App Development: 15:00 - 25:00
  - Discussion on the app development stages and current
progress.

### Text
{{TEXT}}
```

Figure 10: The timeline overview prompt.

A More Details about Prompts

A.1 Summarization Prompting

1) Extractive Summarziation: The extractive summarization prompt is designed to distill the essence of the provided text by identifying and retaining the most significant sentences from the original content. The prompt ensures that the summary is constructed using exact sentences from the text, with minimal modifications for clarity.

2) Abstractive Summarziation: The abstractive summarization prompt instructs the AI to generate concise summaries by interpreting and rephrasing the original text. This method captures the main themes and insights, producing a more human-readable summary.

3) Timeline Summarization: The timeline summarization prompt guides the LLM to create a chronological representation of events or topics discussed during a meeting. This prompt involves identifying the main topics, determining their duration, and generating a visual timeline that highlights the sequence of these discussions. Each segment of the timeline includes brief descriptions, providing a clear and structured overview of the meeting content.