A Retrieval-Augmented Framework For Meeting Insight Extraction

Sartaj Bhuvaji
sbhuvaji@seattleu.edu
Seattle University
Seattle, USA

Prachitee Chouhan
pchouhan@seattleu.edu
Seattle University
USA

Madhuroopa Irukulla
mirukulla@seattle.edu
Seattle University
USA

Jay Singhvi
jsinghvi@seattleu.edu
Seattle University
USA

Wan D. Bae
baew@seattleu.edu
Seattle University
USA

Shayma Alkobaisi
shayma.alkobaisi@uaeu.ac.ae
United Arab Emirates University
United Arab Emirates

Abstract

Meetings are vital for collaboration and decision-making in professional environments, yet recalling key details from past discussions can be challenging and this impacts productivity. In this paper, we address this issue by developing a solution that extracts crucial insights from historical meeting records using Retrieval Augmented Generation (RAG) techniques. Users can easily upload meeting records and query for relevant information. A core feature of our proposed system is grouping meetings based on abstractive summaries, using state-of-the-art clustering algorithms extensively trained for accuracy. Upon user inquiry, the system identifies the most relevant cluster and retrieves related conversations from the Pinecone vector store database. These conversations, paired with custom prompts, are processed through a Large Language Model (LLM) to generate precise responses. Our optimization efforts focus on exploring various encoders and LLMs, with fine-tuning to ensure seamless integration and high performance. This approach tackles challenges in meeting summarization, content discovery, and user-friendly information retrieval.

CCS Concepts

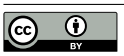
• **Information systems** → **Information retrieval**; • **Computing methodologies** → **Machine learning**.

Keywords

meeting data retrieval, abstractive summarization, speech to text conversion, text summarization, LLM, BART, Pinecone

ACM Reference Format:

Sartaj Bhuvaji, Prachitee Chouhan, Madhuroopa Irukulla, Jay Singhvi, Wan D. Bae, and Shayma Alkobaisi. 2025. A Retrieval-Augmented Framework For Meeting Insight Extraction. In *The 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25), March 31-April 4, 2025, Catania, Italy.* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3672608.3707915



SAC '25, March 31-April 4, 2025, Catania, Italy

ACM ISBN 979-8-4007-0629-5/25/03
https://doi.org/10.1145/3672608.3707915

1 Introduction

In today's dynamic corporate environment, meetings are a daily occurrence, filled with valuable discussions and decisions. However, despite this large volume of valuable insights exchanged, professionals often struggled to recall specific details from past meetings and this hinders productivity and decision-making. The frequency and face-paced and dynamics of meetings make it challenging to capture and access critical information effectively. For instance, imagine a meeting several months ago where a software engineering team debated the ideal user interface design - team members may find it difficult to recall the outcome of that discussion. Managing multiple meeting outcomes can be overwhelming and it often requires excessive searching and referencing.

In this work, we address this challenge by providing a comprehensive solution that seamlessly organizes, summarizes, clusters, and retrieves essential information from historical meeting records. This empowers professionals to navigate and retrieve key information from past meetings. By utilizing innovative technology, we propose a framework that not only organizes meeting content but also facilitates quick and intuitive access to critical decisions and discussions. It will also ease the handling of diverse data, from concise daily discussions to extensive, multi-hour strategy sessions. Our motivation stems from the belief that streamlining this process will enhance productivity, collaboration, and ultimately, the success of projects in today's fast-paced professional environment.

We design and develop a comprehensive system based on Retrieval Augmented Generation (RAG) architecture [10] and powered by Large Language Models (LLMs). The primary objectives include facilitating seamless conversations, recording meetings, summarizing meeting content, and uncovering insightful information. The system is envisioned to present these functionalities through a user-friendly chat interface. The meetings will be clustered once they have been uploaded. Upon querying, relevant data will be transferred to LLM depending on clustering.

To enhance system performance, we investigate the integration of multiple encoders and various LLMs. A clustering-based approach facilitates the ease of reporting semantically similar meetings, which improves retrieval efficiency. Furthermore, fine-tuning is performed on a custom dataset to systematically evaluate and identify the most effective model for seamless system integration. This comprehensive framework addresses key challenges in conversation management, meeting recording, summarization, and

insightful content discovery, ultimately providing a tailored and high-performing solution for users. The proposed methods are extensively tested using audio recordings of real-world professional meetings across four distinct topics.

Our work enhances access to vital meeting information and supports informed decision-making through three key contributions:

- We developed a system framework that allows users to upload meetings, which are transcribed and diarized using AWS Transcribe [1]. The transcripts serve as the foundation for storing data in the Pinecone vector database [12] and generating abstract summaries.
- Abstractive summaries are generated via the BART [9] summarizer that enables the system to cluster meetings with similar content. This clustering improves the organization and accessibility of relevant information.
- Users can query the system through GPT-3.5 Turbo and LangChain [8]. The system then retrieves relevant transcripts from the clustered database through optimization of memory usage and delivers precise, contextually relevant responses to user queries.

2 Related Work

The transformer encoder-decoder architecture has been widely used in natural language processing (NLP) and related applications and various approaches have been discussed for integrating LLMs in the development of a large system. Below, we discuss key components that are relevant to our proposed framework.

Bidirectional and Auto-Regressive Transformer (BART) developed by Facebook is a powerful model for various NLP tasks, such as text generation, translation, classification, and summarization [9]. BART is pretrained by corrupting input text using noise schemes and learning to reconstruct the original input. It reads corrupted text in both directions (left-to-right and right-to-left). The bidirectional encoder produces hidden states that capture the meaning and context of the input, which are then passed to an autoregressive decoder. The decoder generates output one element at a time, conditioning each element on the previously generated ones.

Another key approach that enhances LLMs involves using a vector store for permanent knowledge supplementation. The vector store stores embeddings based on their proximity, which can be retrieved according to user queries. The work in [4] describes how this approach extracts knowledge from the vector store and injects it into the LLM using a prompt and hence enables the model to answer questions it was not specifically trained for.

In contrast to BART, Retrieval-Augmented Generation (RAG) demonstrates more controlled and interpretable hallucinations in question-answering tasks [10] and it becomes preferable for applications requiring more reliable information retrieval. The authors in [11] propose a framework that uses the BART pretrained model to generate abstractive summaries from audio meetings to texts conversations. However, their system is limited to producing summaries for individual meetings, and converting audio/text into diarized transcripts is time-consuming.

Automatic Speech Recognition (ASR) has also seen advancements. Whisper [17], developed by OpenAI, is a state-of-the-art ASR system that converts spoken language into text [14]. It has been trained on a large multilingual, multitask dataset and demonstrates high accuracy and fast transcription capabilities.

For automatic summarization of speech-to-text and speech-to-speech tasks, [6] introduces a two-stage approach. Their method first extracts key sentences and then compresses them using word-based techniques. This strategy balances linguistic and grammatical accuracy with information content and confidence measures, showing strong performance in summarizing spontaneous presentations. Our approach to summarizing spoken dialogues draws on their methodology.

In document clustering, bisecting K-means is shown to outperform agglomerative hierarchical clustering in [16]. Bisecting K-means has superior linear run-time performance and consistently good clustering quality. In contrast, agglomerative clustering struggles when nearest neighbors belong to different classes, resulting in irreversible mistakes. The work in [5] presents a text clustering method based on a semantic similarity graph. The authors use SentenceTransformer for text-to-vector conversion, followed by smoothing embeddings using polynomial filters based on semantic similarity graphs, and then perform clustering with K-means. They evaluate their method using adjusted mutual information and adjusted rand index metrics.

Vector embeddings play a crucial role in NLP tasks such as clustering and text classification. In [16], the selection of Pinecone as a vector database is explained and its efficiency in storing and retrieving vector embeddings. Similarly, [15] provides a summary of models tested using the MTEB benchmark and highlights all-mpnet-base-v2 and all-MiniLM-L6-v2 as optimal models balancing speed and performance, ensuring the selection of the most suitable vector database for various NLP applications.

3 Proposed Framework

3.1 System Overview

In this paper we propose a system architecture that incorporates several new components to enhance retrieval augmentation generation's capabilities. For diarization and abstractive summary, our framework uses Amazon Web Services (AWS) Transcribe, which significantly reduces time for retraining the models. Secondly, a clustering-based solution is implemented to automatically group meetings based on abstractive summary generated using Facebook-BART pre-trained model, accommodating multiple meetings compared to the single-meeting approach proposed in [11]. Next, full transcripts are stored in the Pinecone vector database, rather than abstractive summaries, which is a crucial aspect for the chatbot to effectively answer queries about specific meetings. To retrieve conversations within a window, the relevant top-k vectors are returned by the vector database after recognizing similar meetings using a clustering algorithm, ensuring contextual relevance. Lastly, we fine-tune selected LLMs to use in a chatbot that is capable of handling queries related to meetings.

The framework provides two main use cases: (1) When the user uploads a new meeting in the system, the engine of text diarization and trascription executes, and it is followed by BART Summarizer to generate an abstractive summary. The cluster of this meeting is then identified by a clustering method and all meeting information is stored in a Pinecone database, (2) When the user submits a

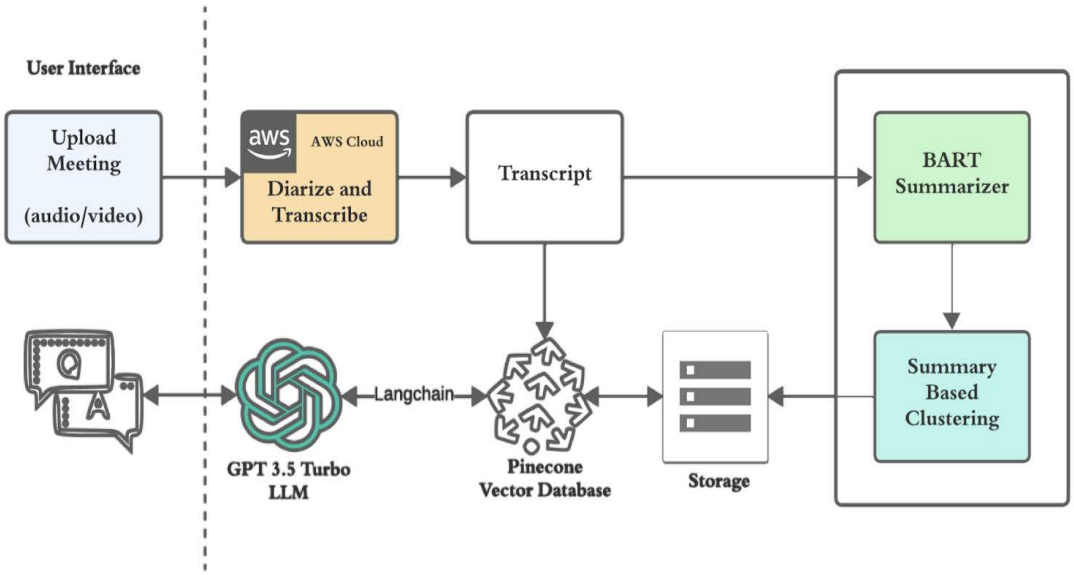


Figure 1: System Framework

query, the relevant meetings are fetched from the database based on the cluster. Further exploration for pertinent answers within these transcript(s) is conducted using the Pinecone database. In this work, to ensure the model generates the diversity and relevance of responses during conversational tasks, the system incorporates techniques such as a Delta parameter in conjunction with Top-k. Subsequently, this conversational frame is provided to the LLM for generating a response to the user's query. An overview of the system is illustrated in Figure 1 and the detailed tasks are explained below.

3.1.1 Meeting uploading.

- *Speech-to-Text Conversion*: Using AWS Transcribe, the video or audio meeting material is first converted into a text transcript and saved in the Pinecone database. Additionally, the transcript employs the Facebook-BART model to provide abstractive summaries, which are subsequently clustered based on the query vector to produce relevant meetings. The resulting summaries are then saved in a JSON file.
- *Clustering modeling*: The text embedding model OpenAI embeddings is used and clusters are formed based on the clustering algorithm. The result of the model is stored and used when the user poses a query.

3.1.2 Chat Interface.

- *LangChain framework*: The LangChain framework is employed to supply pertinent data to LLMs from the Pinecone database. The clustering algorithm sits on top of the Pinecone database and fetches relevant clusters of meetings based on Facebook AI Semantic FAISS semantic search.
- *Response Generation*: The context from meetings which is provided by Pinecone, based on the clustering model, is passed to the LLM which is used to generate answers for the user's query.

3.2 Methods

3.2.1 Clustering Methods. In the proposed framework, two clustering algorithms are applied and their performance analysis is presented in Section 4.

- *Mean-Shift*: A non-parametric algorithm clusters data by iteratively shifting data points towards regions of higher density until convergence. It does not require the number of clusters to be predefined and can handle clusters of irregular shapes.
- *Hierarchical Density-Based Spatial Clustering (HDBSCAN)*: A density-based algorithm finds clusters of varying shapes and densities in high-dimensional datasets. It is effective in managing noise and outliers without needing a preset number of clusters.

Smoothing embedding before clustering models can help better differentiate between similar and dissimilar abstractive summaries, which results in performance gains. In our framework, we utilize the Semantic Graph smoothing method presented in [5]. We use the following polynomial filters on the embedding matrix for smoothing:

(1) Simple Graph Convolution (SGC): SGC simplifies traditional graph convolutions by removing non-linearities and stacking, effectively applying a polynomial filter over the adjacency matrix to smooth node embeddings.
(2) Simple spectral Graph Convolution (S2GC): S2GC uses a spectral polynomial filter to aggregate information from multiple graph powers, improving node embedding by capturing a broader range of node interactions.
(3) Decoupled Graph Convolution (DGC): DGC decouples the feature transformation and propagation steps in graph convolutions, approximating the graph smoothing process with fewer computational resources.

Table 1: Audio meeting datasets

Meeting category	Total number of meetings	Each audio file length range (in minutes)
Architecture	11	12 - 51
Remote Control	4	17 - 38
HR	6	14 - 57
Social Media	2	20 - 27

(4) Approximate personalized propagation of neural predictions (APPNP): APPNP combines personalized PageRank with graph convolutional networks, using a polynomial filter to control the balance between local smoothing and retaining the original node features.

3.2.2 LLM Retraining. The construction of these new components builds upon existing ideas and insights from relevant literature, such as [11] for storing and utilizing meeting conversations and [4] for understanding the importance of fine-tuning LLMs to overcome hallucinations. For clustering and linkage of meetings based on relevance, keywords and summaries are extracted, and custom functions are implemented to query the database for the most relevant conversations. Custom prompts aid in feeding relevant data to the LLM, enhancing its ability to comprehend queries within context. Moreover, the system incorporates a chain of memory to enable the LLM to retain in-chat memory, crucial for referring to context from previous messages. The fine-tuning of LLMs utilizes the Deep Speed framework developed by Microsoft [15], and a comparison of multiple encoding and LLMs is conducted using metrics outlined in [13].

In summary, the proposed system architecture involves a multi-faceted approach, incorporating novel methods and algorithms to improve meeting clustering, diarization, and chatbot responsiveness.

4 Experiments

4.1 Datasets and Evaluation Metrics

We tested on 23 conversational meetings from the AMI Corpus [2] and ICSI Corpus [7], which belong to four category of topics. Details of the meeting datasets are shown in Table 1.

4.1.1 Evaluation metrics for abstractive summarization.

(1) Recall-Oriented Understudy for Gisting Evaluation (ROGUE): A widely used metric for evaluating automatic summarization, measuring the overlap between generated and reference summaries.
   - ROUGE-1: Measures unigram (single word) overlap.
   - ROUGE-2: Measures bigram (two-word) overlap.
   - ROUGE-L: Measures the longest common subsequence, accounting for sentence structure.
   - ROUGE-Lsum: Calculates the ROUGE-L score for all sentences at the document level, rather than averaging individual sentence scores.

(2) BERTScore: It evaluates semantic similarity between a generated summary and a reference text using BERT embeddings and cosine similarity and provides precision, recall, and F_1 scores. Unlike ROUGE, which focuses on exact word matches, BERTScore incorporates semantic understanding and context of the words, enabling a more nuanced evaluation of key ideas and meaning, even when the wording differs from the reference. This makes BERTScore particularly effective for evaluating abstractive summarization models like BART, as it assesses summary quality through contextual embeddings and a deeper understanding of word relationships.

4.1.2 Evaluation metrics for clustering.

(1) Adjusted Rand Index (ARI): ARI measures the similarity between two clusterings, adjusted for chance. It is related to accuracy and applicable even without class labels. A higher ARI score indicates better agreement between the clusterings.
(2) Adjusted Mutual Information (AMI): AMI is a variation of mutual information, adjusted for chance agreement between clusterings, similar to ARI. A higher AMI score reflects better alignment between the clusters.
(3) B-cubed precision and recall: These metrics evaluate clustering quality, particularly for overlapping clusters. Precision measures cluster purity, while recall assesses cluster coverage. They provide a comprehensive assessment of clustering performance and valuable insights for algorithm selection and tuning.

4.1.3 Evaluation metrics for RAG. We evaluate our RAG model using the following metrics:

- Groundedness: Measure of how well the answer is supported by the context.
- Answer Relevance: Measure of how well the answer is relevant to the question.
- Context Relevance: Measure of how well the context fetched from DB is relevant to the question.
- Cosine Similarity: Measure of how similar the answer by LLM is to the ground truth.

4.2 Abstractive Summarization Evaluation

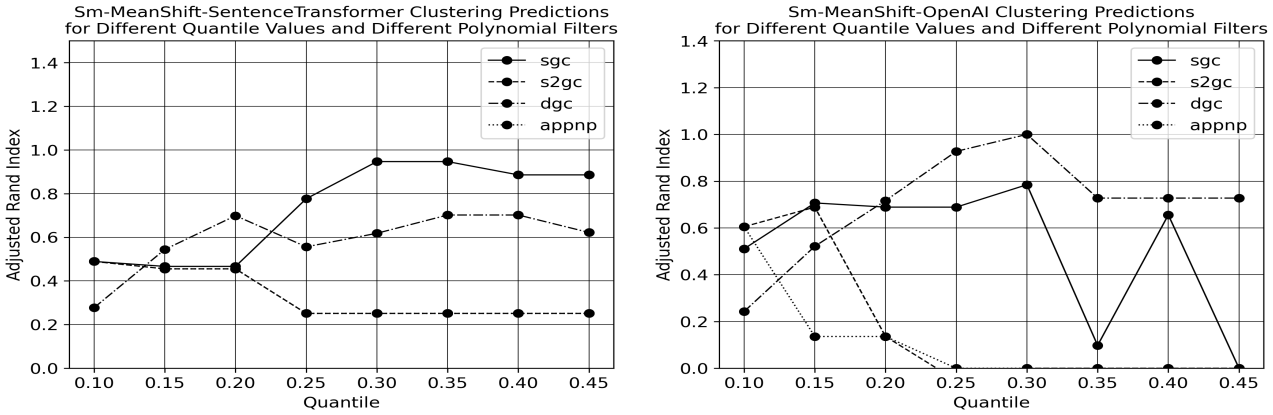
ROGUE scores as shown in Table 2 are relatively low as they primarily focus on measuring the overlap between the words or n-grams in the generated summaries and the reference summaries. This means that ROUGE does not consider the semantic similarity or the context of the words, which can lead to lower scores, especially for complex topics or summaries that are well-written but use different wording. However, in the case of Remote Control, the ROUGE-1 score is 0.521, indicating a relatively high level of overlap for unigrams between the generated summaries and the reference summaries for that topic.

Table 2: ROURGE score for abstractive summarization quality

Meeting category	ROUGE-1	ROURGE-2	ROUGE-L	ROURGE-Lsum
Architecture	0.223	0.026	0.144	0.144
Remote Control	0.521	0.240	0.379	0.379
HR	0.512	0.221	0.378	0.384
Social Media	0.413	0.118	0.307	0.307

Table 3: BERTScore for abstractive summarization quality

Meeting category	Precision	Recall	F_1 score	Similarity score
Architecture	0.584	0.587	0.585	0.864
Remote Control	0.720	0.698	0.709	0.898
HR	0.697	0.724	0.710	0.911
Social Media	0.716	0.705	0.710	0.905



(a) Sentence Transformer

(b) OpenAI

Figure 2: Hyperparameter tuning of Smoothing + Mean-Shift with Sentence Transformer and OpenAI

BERTScores as shown in Table 3 exhibit relatively consistent performance across precision, recall, F1 score, and similarity score for each topic, providing a comprehensive view of the model's performance. The precision and recall values for each topic are relatively balanced, with small differences between them. This suggests that the BART model is able to achieve a good balance between precision and recall for these topics. The similarity scores for all topics are relatively high, ranging from 0.8641 to 0.9114, indicating that the BART model's output is quite similar to the reference summaries.

BERTScore's more semantically accurate evaluation ability can contribute to higher similarity scores compared to ROUGE. Overall, the lower ROUGE scores compared to BERTScores can be attributed to the limited scope of ROUGE in capturing the semantic aspects of summarization.

4.3 Clustering Evaluation

We constructed clustering models based on two algorithms, Mean-Shift and HDBSCAN. We present the perforemance of (1) baseline clustering models (without smoothing) and (2) smoothing + clustering models with two known LLMs, Sentence Transformer and OpenAI.

4.3.1 Hyperparameter training. Hyperparameters were chosen by an extensive training and testing. Base hyperparameters we used in this paper are: min_cluster_size: 2, metric: 'euclidean', and cluster_selection_method: 'leaf'. For clustering models created with semantic graph smoothing, we used the hyperparameters $k = 10$ of the k-NN graph, order of propagation, $degree = 2$, the parameter λ (lamda) =1 and the filter specific parameters α =0.1 and T=5, which were used in [5].

We performed hyperparameter tuning for training and evaluating the data with different quantities. Hyperparameter tuning for Smoothing Mean-Shift with Sentence Transformer and OpenAI are shown in Figure 2.

4.3.2 Vector Embedding: Sentence Transformer vs. OpenAI. Table 4 presents the performance of baseline models across five common metrics. Both Mean-Shift and HDBSCAN clustering algorithms

Table 4: Results of baseline models with Sentence Transformer

Clustering	Adjusted mutual information	Adjusted rand index	B-cubed precision	B-cubed recall	B-cubed F_1
Mean-Shift	0.403	0.368	0.492	0.923	0.641
HDBSCAN	0.465	0.290	0.503	0.860	0.635

Table 5: Results of smoothing + clustering with Sentence Transformer

Clustering	Smoothing filter	Adjusted mutual information	Adjusted rand index	B-cubed precision	B-cubed recall	B-cubed F_1
Mean-Shift	SGC	0.526	0.560	0.833	0.677	0.747
	S2GC	0.350	0.201	1.000	0.503	0.669
	DGC	0.618	0.663	0.781	0.804	0.793
	APPNP	0.350	0.201	1.000	0.503	0.669
HDBSCAN	SGC	0.480	0.371	0.550	0.817	0.658
	S2GC	0.501	0.278	0.525	0.870	0.655
	DGC	0.281	0.154	0.341	0.804	0.479
	APPNP	0.501	0.278	0.525	0.870	0.655

Table 6: Results of Baseline models with OpenAI

Clustering	Adjusted mutual information	Adjusted rand index	B-cubed precision	B-cubed recall	B-cubed F_1
Mean-Shirt	0.324	0.221	0.759	0.621	0.683
HDBSCAN	0.666	0.518	0.619	0.942	0.747

Table 7: Results of smoothing + clustering with OpenAI

Clustering	Smoothing filter	Adjusted mutual information	Adjusted rand index	B-cubed precision	B-cubed recall	B-cubed F_1
Mean-Shift	SGC	0.603	0.603	0.833	0.735	0.780
	S2GC	0.000	0.000	1.000	0.388	0.559
	DGC	0.841	0.898	0.913	0.884	0.893
	APPNP	0.000	0.000	1.000	0.388	0.559
HDBSCAN	SGC	0.501	0.329	0.517	0.882	0.652
	S2GC	0.666	0.518	0.619	0.942	0.747
	DGC	0.488	0.270	0.485	0.913	0.634
	APPNP	0.666	0.518	0.619	0.942	0.747

show relatively low scores in terms of Adjusted Mutual Information (AMI) and Adjusted Rand Index (ARI), which indicates weaker alignment between the predicted clusters and the ground truth when applied without additional techniques.

In Table 5, we observe the impact of semantic graph smoothing on Mean-Shift and HDBSCAN clustering. These results, based on optimized parameters, show improvements in clustering performance, although the gains are more pronounced for some filters than others.

Next, we present the results of baseline models trained with OpenAI embeddings, as presented in Table 6. While Mean-Shift continues to show low AMI and ARI scores, HDBSCAN demonstrates a noticeable improvement when trained with OpenAI embeddings compared to those trained with Sentence Transformer embeddings.

Table 7 shows the results of Mean-Shift and HDBSCAN after applying semantic graph smoothing with OpenAI embeddings. The Mean-Shift algorithm, combined with the DGC filter, outperforms other models across all evaluation metrics. This demonstrates the benefits of semantic graph smoothing in enhancing cluster quality.

While Mean-Shift with the S2GC and APPNP filters shows lower ARI, other models exhibit significant improvements, particularly in ARI scores. This suggests that OpenAI embeddings contribute to forming more robust and accurate clustering models and enhance their ability to identify clusters that are relevant to user queries. These findings highlight the potential of OpenAI embeddings in building effective clustering frameworks for real-world applications.

4.4 Retrained LLM Evaluation

4.4.1 Comparing LLMs. In this study, we evaluate and compare the performance of four popular LLMs: GPT-3.5 Turbo, Gemini Pro 1.0, Mistral Large, and Claude 3 Opus. These models were tested on a set of datasets comprising twenty-three meetings, each with

Table 8: Sample groundedness, context relevance, answer relevance, and cosine similarity scores

No	User Input	Response	Groundedness	Context Relevance	Answer Relevance	Cosine Similarity
1	Who recommends universal masking?	The American Academy of Pediatrics and CDC recommends universal masking.	1	0.9	1	1
2	How many patients are in the ICU Unit?	There are 23 patients in the ICU Unit located at the East Wing.	1	0.7	1	0.96
3	What is the capital of France?	Paris is the capital of France.	0	0	1	0

five sample questions. The meetings are related to COVID-19, with participants primarily consisting of medical staff at a hospital.

We retrained the GPT-3.5 Turbo model and evaluated its performance across various LLM evaluation metrics. Table 8 provides an example, illustrating how the models responded to specific questions. For instance, the first two questions received high scores across all metrics as they were directly grounded in the meeting content. However, for a question like "What's the capital of France?"—a topic unrelated to the meeting—the model still provided an answer, highlighting the issue of LLM hallucination. To mitigate this, we applied prompt engineering techniques, refining the model's ability to remain grounded in context.

The results of all four models are presented in Figure 3. We observed relatively low groundedness scores across the models, primarily because LLMs are designed to generate coherent and fluent text, which can sometimes sacrifice factual accuracy. Claude 3 Opus delivered the highest scores for answer relevance, which may be attributed to its larger number of parameters, allowing for more precise and contextually appropriate responses.

In terms of cosine similarity (measuring how closely the generated answer matches the ground truth), the models scored between 0.6 and 0.8. While higher values indicate closer matches to the reference answers, cosine similarity is sensitive to word order. Even if an LLM produces a correct answer, variations in phrasing can lead to lower scores.

Overall, Claude 3 Opus and GPT-3.5 Turbo emerged as the best performers for our use case, balancing fluency, coherence, and factual grounding more effectively than the other models.

4.4.2 Comparing LLM Memory. Conversational memory allows a chatbot to maintain context over multiple queries, enabling coherent conversations. Without it, each query would be treated as an independent input, ignoring prior interactions. Memory enables LLMs to retain previous exchanges with users. By default, however, LLMs are stateless, meaning each query is processed in isolation.

In our experiments, we tested four memory options:

(1) ConversationalBufferMemory: Stores all interactions in sequence within the memory buffer.
(2) ConversationalBufferWindowMemory: Stores only the last n interactions in the memory buffer.
(3) ConversationalSummaryMemory: Stores a condensed summary of past interactions, capturing key details to maintain context and improve continuity.

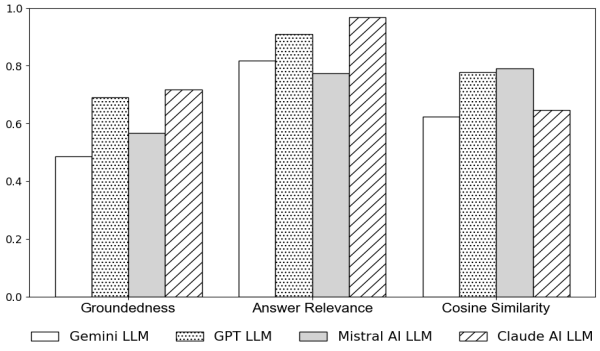


Figure 3: Comparing LLM models

(4) ConversationSummaryBufferMemory: Summarizes the message history into n tokens and stores the summary in the memory buffer.

We found that ConversationSummaryBufferMemory effectively mitigates the risk of exceeding the LLM's token limit by summarizing previous interactions. For this reason, we adopted ConversationSummaryBufferMemory with a 650-token summary in our experiments. The results of our memory experiments are shown in Figure 4.

5 Conclusion

In this paper, we proposed a system framework that addresses the critical challenge of managing and retrieving valuable insights from past meetings, which is a common issue in today's fast-paced corporate environment. By utilizing a comprehensive architecture built on Retrieval Augmented Generation and powered by LLMs, we developed a system that not only organizes and clusters meeting transcripts but also allows for intuitive and efficient retrieval of relevant information. The use of AWS Transcribe for diarization and transcription, combined with BART for abstractive summarization and Pinecone for vector storage, ensures that meeting content is accessible, organized, and clustered for optimal retrieval. Furthermore, our integration of GPT-3.5 Turbo with LangChain provides users with an interactive, query-based interface to extract essential information quickly and accurately. This solution enhances productivity, streamlines decision-making, and reduces the cognitive load

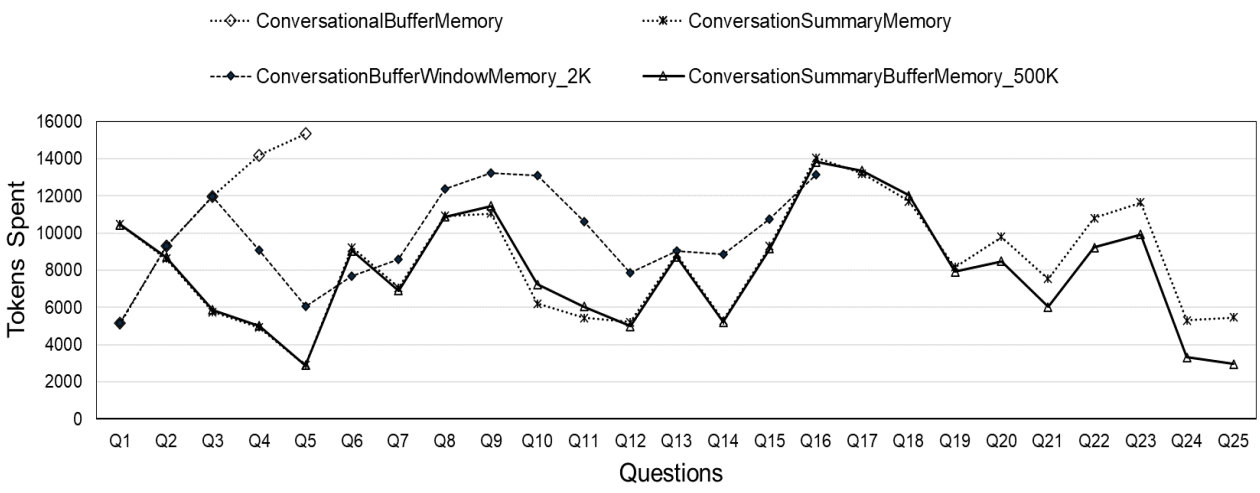


Figure 4: Comparing memory consumption for varying memory types

on professionals, ensuring that vital discussions are easily accessible and actionable, regardless of the frequency or complexity of their meetings.

Future work can include fine-tuning with QLoRA [3] to improve relation extraction and response accuracy, researching faster transcription models for quicker speech-to-text conversion, and exploring different embedding models for better contextual understanding. We will also investigate open-source LLMs to enhance language generation and reduce costs. Additionally, video frame tagging will be explored to extract and tag key points from meeting recordings, providing visual summaries alongside insights.

Acknowledgments

This work was supported in part by United Arab Emirates University under UAEU-NFRP Grant No. G00004281 form United Arab Emirates University, and in part by Seattle University under Bannan Endowed Chair of Engineering Award.

References

[1] Inc. Amazon AWS. accessed in September, 2024. AWS Subscribe. https://aws.amazon.com/pm/transcribe/

[2] European Commission. accessed in 2023. AMI Corpus. https://groups.inf.ed.ac.uk/ami/corpus/

[3] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems* 36 (2024).

[4] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. RAGAS: Automated Evaluation of Retrieval Augmented Generation. arXiv:2309.15217 [cs.CL]

[5] Chakib Fettal, Lazhar Labiod, and Mohamed Nadif. [n. d.]. More Discriminative Sentence Embeddings via Semantic Graph Smoothing. ([n. d.]).

[6] S. Furui, T. Kikuchi, Y. Shinnaka, and C. Hori. 2004. Speech-to-text and speech-to-speech summarization of spontaneous speech. *IEEE Transactions on Speech and Audio Processing* 12, 4 (2004), 401–408. https://doi.org/10.1109/TSA.2004.828699

[7] UC Berkeley International Computer Science Institute Speech Group. accessed in 2023. ICSI Corpus. https://groups.inf.ed.ac.uk/ami/icsi/

[8] LangChain.com. accessed in September, 2024. LangChain. https://www.langchain.com

[9] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 9459–9474. https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf

[11] Vincent Marklynn, Anjali Sebastian, Yong Long Tan, Wan D. Bae, Shayma Alkobaisi, and Sada Narayanappa. 2024. A Framework for Abstractive Summarization of Conversational Meetings. In *Proceedings of the 14th IEEE Annual Computing and Communication Workshop and Conference (CCWC 2024)*. 507–512.

[12] Inc. Pinecone Systems. accessed in September, 2024. Pinecone Vector Database. https://www.pinecone.io

[13] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and Narrowing the Compositionality Gap in Language Models. arXiv:2210.03350 [cs.CL]

[14] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356* (2022).

[15] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3505–3506.

[16] Michael Steinbach. 2000. *A Comparison of Document Clustering Techniques*. Technical Report. Technical Report# 00_034/University of Minnesota.

[17] Whisper. accessed in September, 2024. Whisper. https://github.com/openai/whisper