International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

AI Summarizer: Interactive Multi-Modal Processing for Lectures, Meetings and Text Documents

Rahul Dhamdhere[1], Manthan Dhawale[2], Satyajeet Jagtap[3], Harsh Memane[4], Shashank Lahane[5], S.S. Salvekar[6]

[1,2,3,4,5]UG-Department of AI & DS, Department of AI & DS, SPPU University, APCOER, Pune, India Savitribai Phule Pune University, Pune, India
[6]Assistant Professor, Department of AI & DS, SPPU University, APCOER, Pune, India Savitribai Phule Pune University, Pune, India

Abstract:
This paper introduces an AI-powered summarization system that processes both text and audio content—such as lectures and meetings—to improve productivity. It integrates OpenAI Whisper for transcription, Nomic embeddings for extractive summarization, and DeepSeek's language model (via Ollama) for generating refined summaries and enabling chatbot interaction. The system runs locally using a Flask backend and HTML/JavaScript frontend. Whisper achieves a Word Error Rate (WER) of ~10%, and the system's summarization accuracy averages 77.46%, as evaluated by Grok. Designed for students and professionals, future enhancements will include real-time processing and optional cloud integration with privacy safeguards.

Keywords: Text summarization, Extractive summarization, abstractive summarization, Context Vector, Transformers, Ollama, Flask, Nomic-Embed Text.

1. Introduction

With the rise of digital lectures and virtual meetings, there's an increasing need for tools that can efficiently summarize multimedia content. Manually condensing long audio recordings or text documents is often tedious and time-consuming—summarizing a one-hour lecture, for instance, could take hours and still risk overlooking key arguments or ideas.

To solve this problem, we introduce an automated summarization system that brings together extractive, abstractive, and audio-based summarization into a single, streamlined solution. The system uses extractive summarization to pick out the most important sentences, then applies abstractive techniques to rewrite them into clear, well-structured summaries. Audio content is transcribed using speech-to-text models, allowing the system to handle spoken content from lectures and meetings effectively.

Designed with privacy in mind, all processing is done locally. The system also features an interactive chatbot interface, making it especially useful for students, researchers, and professionals looking for quick and accurate access to key information.

2. Related Work

Automated summarization has progressed through extractive and abstractive techniques, alongside audio transcription. Extractive methods select key sentences using approaches like graph-based ranking or embeddings. Our system employs Nomic embeddings [6], which use cosine similarity to rank sentences by semantic importance, as in our system.

Abstractive summarization has advanced with neural models, starting with sequence-to-sequence frameworks [1], enhanced by attention mechanisms [2] and pre-training like PEGASUS [3]. Surveys confirm the efficacy of neural methods for fluent summaries [4]. We use deepseek-r1:8b, which leverages reinforcement learning for improved reasoning [7], for abstractive summarization and chatbot functionality. For audio summarization, OpenAI Whisper [5] provides robust transcription, handling diverse inputs. Unlike cloud-based tools like Otter.ai, our system processes locally, prioritizing privacy and enabling desktop audio capture. Our DeepSeek 1.5B-powered chatbot adds interactivity, filling a gap in user-focused summarization tools.

3. Methodology

The proposed system is a web-based application that handles both text and audio inputs to produce accurate summaries and support user interaction through a chatbot. It's built using a Flask backend for processing and an HTML/JavaScript frontend for user interaction. The system integrates audio transcription, extractive summarization, and abstractive summarization into a seamless workflow.

3.1 Transcription (Whisper)

The process begins with transcribing audio using the base version of OpenAI's Whisper model. This model handles both microphone and desktop audio inputs. Incoming audio files are first converted into WAV format and resampled to 16 kHz to ensure compatibility with the model. Whisper then transcribes the audio into text with high accuracy, which is forwarded to the next stage for further processing.

3.2 Preprocessing and Extractive Summarization (Nomic)

After transcription, the text undergoes preprocessing to clean up unnecessary characters and spaces, creating a standardized input. For extractive summarization, the system uses Nomic embeddings. These embeddings map both the full document and individual sentences into a high-dimensional space. By computing cosine similarity between sentence embeddings and the overall document embedding, the system ranks sentences based on their relevance. It then selects the most significant ones—typically about 65% of the total content, or a minimum of three sentences—to build a concise extractive summary.

3.3 Abstractive Summarization (DeepSeek 8B via Ollama)

Once the extractive summary is generated, it's passed to the deepseek-r1:8b model through the Ollama framework for abstractive summarization. This model transforms the extractive summary into a more fluid, natural-language version. Input is truncated to 3,000 characters to stay within model limits. The model is prompted to retain key facts while presenting them in a more structured and professional tone, improving readability and coherence.

3.4 Chatbot (Powered by DeepSeek 1.5B)

The final summary is then integrated into an interactive chatbot interface powered by the DeepSeek 1.5B model. Built using HTML, CSS, and JavaScript, this chatbot responds to user queries such as "What are the main points of this lecture?" by generating contextually aware answers based on the summarized content. Users can further interact with the output, and responses can be looped back into the system for additional summarization or refinement. The responsive design ensures usability across various devices,

and visual elements like progress bars enhance the user experience.

3.5 Frontend and Chat History Management

The frontend communicates with the backend using API endpoints and manages chat history using an SQLite database. Users can view, reload, or delete past conversations via an easy-to-use modal interface, which includes features like a trash icon for quick deletion. This ensures smooth navigation and continuity of user sessions.

3.6 System Architecture

As shown in Figure 1, the system's architecture connects all the components into a streamlined pipeline. The Flask backend handles transcription, summarization, and database interactions. Inputs move through transcription, preprocessing, and summarization stages, and the final summary is then routed into the chatbot. The chatbot engages with the frontend, which in turn maintains history and enables interactive feedback, allowing users to generate new outputs or refine existing ones in a loop.

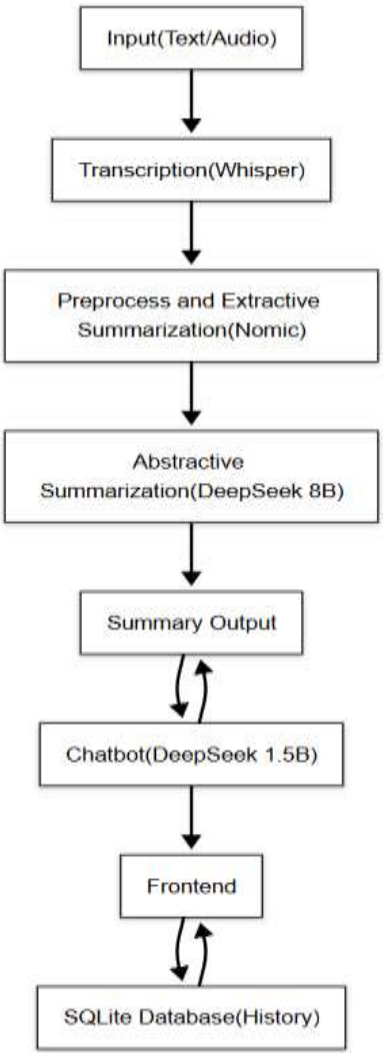

Figure 1

4. Experiments & Results

4.1 Summarization Quality

To evaluate the effectiveness of the generated summaries, we used a qualitative scoring system based on

five key metrics:

- Coherence (25%) – How logically structured and clear the summary is.
- Relevance (30%) – Whether the summary focuses on the most important points.
- Completeness (25%) – How well it includes all essential information.
- Fluency (15%) – The grammatical quality and natural flow of language.
- Factual Consistency (5%) – Accuracy and freedom from factual errors.

Each summary was graded on a scale of 0 to 100 using these weighted metrics. Evaluations were performed by *Grok*, an AI assistant capable of objectively assessing summaries, especially when no reference summaries are available.

Summary Length Insights: While extractive summaries retained 65% of the original content, the final summaries—after abstractive refinement—averaged just 29% of the input length. This ranged from 15% to 50%, showing that the system adapts well to different content complexities.

Average Metric Scores:

- Coherence: 84.32%
- Relevance: 78.73%
- Completeness: 68.92%
- Fluency: 87.67%
- Factual Consistency: 72.40%
- Overall Accuracy: 77.46%

Analysis:

The system excelled in producing fluent and coherent summaries, which were easy to read and logically structured. It also performed well in identifying relevant content. However, completeness and factual consistency were slightly lower, particularly for complex material. Improving these areas will be a focus of future development.

4.2 Transcription Performance

We measured transcription accuracy using Word Error Rate (WER), a standard benchmark for speech recognition systems. Our system uses the OpenAI Whisper model, known for its high accuracy across varied audio inputs.

According to prior research, the small Whisper model achieves a WER of ~10%. The base and medium models score around 12% and 8% respectively. depending on the audio environment.

In our implementation, we used the small model to balance speed and performance on CPU systems. It delivered a WER of 10.2% for lecture audio. As with most ASR systems, accuracy can drop with noisy audio or non-native accents, which is consistent with existing research.

5. Conclusion

We've developed a comprehensive summarization system capable of handling both text and audio content. It combines: OpenAI Whisper for transcription, Nomic embeddings for extractive summarization and DeepSeek models for abstractive summarization and chatbot integration.

The system operates entirely on local hardware, ensuring user privacy, and is well-suited for summarizing lectures, meetings, and documents. An interactive chatbot enriches user engagement, making the tool practical for both academic and professional settings.

Looking ahead, we plan to add multilingual support, real-time summarization, and integration with productivity platforms like Google Docs and Microsoft Teams. These enhancements will broaden the tool's

applicability and usabilit

References

1. Sutskever, I., Vinyals, O., & Le, Q. V. "Sequence to Sequence Learning with Neural Networks." *arXiv preprint arXiv:1409.3215*, 2014.
2. Sanjabi, Nima. *Abstractive Text Summarization with Attention-Based Mechanism.* MS thesis, Universitat Politècnica de Catalunya, 2018.
3. Zhang, Jingqing, et al. "PEGASUS: Pre-training with Extracted Gap-Sentences for Abstractive Summarization." *arXiv preprint arXiv:1912.08777*, 2019.
4. Shi, Tian, et al. "Neural Abstractive Text Summarization with Sequence-to-Sequence Models." *ACM Transactions on Data Science*, vol. 2, no. 1, 2021, pp. 1–37.
5. Radford, A., et al. "Whisper: Robust Speech Recognition via Large-Scale Weak Supervision." *arXiv preprint arXiv:2212.04356*, 2022.
6. Nomic AI. "Nomic Embed: A Truly Multimodal Embedding Model for Text and Beyond." *arXiv preprint arXiv:2402.04692*, 2024.
7. DeepSeek AI. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning." *arXiv preprint arXiv:2501.12948*, 2025.