International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI-Based Podcast Summarizer & Keyword Extractor

Prof Priyanka V [1]*, Adarsh HR*[2]*, Bangaru Sathwik Reddy*[3]*, Harsha Gowda M*[4]*, Mukesh S*[5]

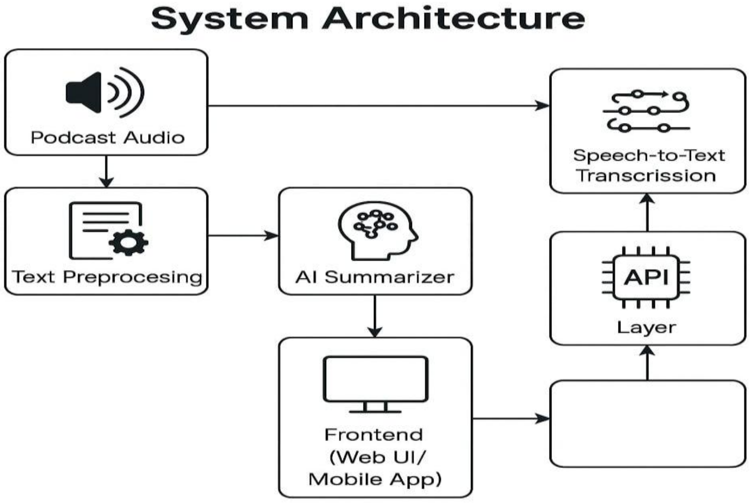
[1]Professor, Department of Computer Science and Engineering (Artificial Intelligence), DSATM, Bangalore, India.
[2]Student (USN: 1DT22CA002), Department of Computer Science and Engineering (Artificial Intelligence), DSATM, Bangalore, India.
[3]Student (USN: 1DT22CA009), Department of Computer Science and Engineering (Artificial Intelligence), DSATM, Bangalore, India.
[4]Student (USN: 1DT22CA021), Department of Computer Science and Engineering (Artificial Intelligence), DSATM, Bangalore, India:
[5]Student (USN: 1DT22CA028), Department of Computer Science and Engineering (Artificial Intelligence), DSATM, Bangalore, India.

Abstract:

In recent years, podcasts have become a dominant medium for sharing knowledge, entertainment, and discussions across various domains. However, their long-form, unstructured nature makes it difficult for listeners to extract key insights quickly. This paper introduces, an AI-driven system designed to automatically summarize podcast episodes and extract relevant keywords. integrates speech-to-text transcription, natural language summarization, and keyword extraction to transform lengthy audio content into concise, readable summaries. Using transformer-based models and advanced NLP techniques, the system improves content accessibility, enhances user experience, and supports content indexing. This research demonstrates the feasibility and effectiveness of automated podcast summarization, offering a scalable solution for users and content platforms alike.With the increasing popularity of podcasts as a medium for sharing information, education, and entertainment, users often face the challenge of navigating through lengthy, unstructured audio content to extract meaningful insights. Unlike written articles or videos with embedded transcripts, podcasts generally lack intuitive summarization or keyword-based indexing, making it difficult for listeners to preview or revisit essential points. This paper presents Echomind, an AI-driven podcast summarization and keyword extraction system designed to address this problem. Echomind leverages a combination of automatic speech recognition (ASR), natural language processing (NLP), and transformer-based summarization models to convert raw audio into concise, human-readable summaries. Additionally, the system identifies and extracts key phrases that represent the core topics discussed in each episode. The result is an efficient, scalable solution that improves podcast accessibility, enhances content discovery, and supports both end-users and platform developers. Experimental results demonstrate Echomind's capability to maintain high-quality summarization performance while significantly reducing the time needed to understand podcast content.

Introduction:

Podcasts have surged in popularity, offering rich, long-form content across education, business, health, and entertainment. Despite their value, podcasts present a challenge: users often lack the time or patience to listen to entire episodes. Unlike text-based media, audio content is harder to skim, search, and categorize. This limits discoverability and deters potential listeners who prefer concise formats. To address these issues, this paper proposes Echomind, an AI-powered podcast summarizer and keyword extractor. Echomind converts raw audio into a structured text summary with key topic highlights, helping users quickly determine the relevance of episodes. The system combines automatic speech recognition (ASR), abstractive summarization, and keyword extraction to provide a seamless solution. Echomind enhances user experience, supports metadata generation for platforms, and enables better indexing of audio content for search engines. Podcasts have rapidly become one of the most influential forms of media, offering long-form, conversational content across diverse fields such as science, education, business, and entertainment. According to recent trends, millions of podcast episodes are produced annually, attracting listeners globally. However, the very strength of podcasts—their in-depth and free-flowing nature—also presents a challenge: they are time-consuming and difficult to scan or search efficiently. For users seeking quick information or trying to evaluate a podcast's relevance, the traditional listen-through approach is impractical. This research introduces Echomind, a comprehensive AI-based system designed to automatically summarize podcasts and extract relevant keywords. The system combines cutting-edge components including speech-to-text conversion using Automatic Speech Recognition (ASR), abstractive summarization using transformer-based deep learning models, and keyword extraction powered by NLP algorithms. Echomind aims to bridge the gap between long-form audio content and the need for fast, efficient information consumption. The goal of this study is not only to design and implement such a system but also to evaluate its performance and effectiveness in real-world podcast use cases. By improving the accessibility and usability of podcast content, Echomind contributes to the broader vision of AI-enhanced media interaction.



System Architecture

The architecture of Echomind is designed as a modular pipeline, enabling seamless processing of podcast audio from raw input to summarized output with keyword extraction. The system is divided into four major components: (1) Audio Ingestion, (2) Speech-to-Text Conversion, (3) Text Summarization, and (4) Keyword Extraction. Each module operates independently yet communicates in sequence to maintain end-to-end efficiency and accuracy.

3.1 Audio Ingestion Module

The first step in the Echomind pipeline involves accepting input audio files in standard formats such as MP3, WAV, or FLAC. This module handles pre-processing tasks such as:

- Downsampling and normalizing audio
- Removing background noise (using filters or libraries like PyDub or Librosa)
- Segmenting long audio into smaller, manageable chunks for efficient processing

This stage ensures compatibility and optimal input quality for the subsequent ASR system.

3.2 Speech-to-Text Conversion

This module utilizes state-of-the-art Automatic Speech Recognition (ASR) to transcribe audio into text. For this purpose, Echomind integrates OpenAI's Whisper due to its robustness in multilingual transcription, noise tolerance, and high accuracy on conversational speech. Key steps include:

- Feeding cleaned audio segments into the Whisper model
- Performing speaker diarization (optional) to distinguish multiple speakers
- Reconstructing the full transcript in logical order

The output of this stage is a raw text transcription that forms the input for summarization and keyword extraction.

3.3 Text Summarization Engine

The core of Echomind's functionality lies in its ability to generate coherent and human-like summaries. This is achieved using transformer-based abstractive summarization models, such as BART or T5. These models are fine-tuned to understand conversational language and long-form discourse. The summarization engine includes:

- Chunking long transcripts to fit model input size
- Context-aware summarization for each segment
- Merging segment summaries into a final coherent output

This component ensures that the main ideas, speaker intent, and contextual flow of the episode are preserved while reducing the content length significantly.

3.4 Keyword Extraction Module

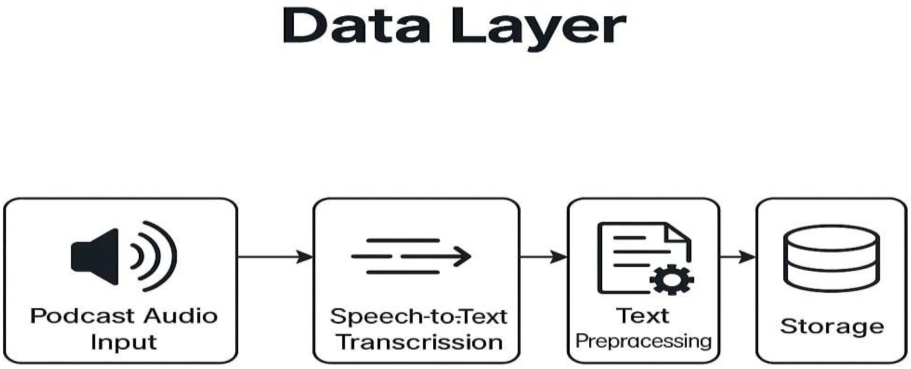
To complement the summary, Echomind uses NLP-based techniques to extract keywords that reflect the main themes of the episode. The keyword extraction component leverages:

- TF-IDF for basic statistical keyword selection
- KeyBERT for embedding-based relevance scoring
- Named Entity Recognition (NER) for identifying proper nouns, organizations, and key entities

The result is a set of topic-representative keywords that enhance content discoverability, tagging, and searchability.

3.5 Integration and User Interface

All components are integrated into a lightweight backend powered by **Flask** (or FastAPI) and connected to a front-end user interface where users can upload audio, view summaries, and download outputs. The modular design ensures scalability and easy updates to individual components.

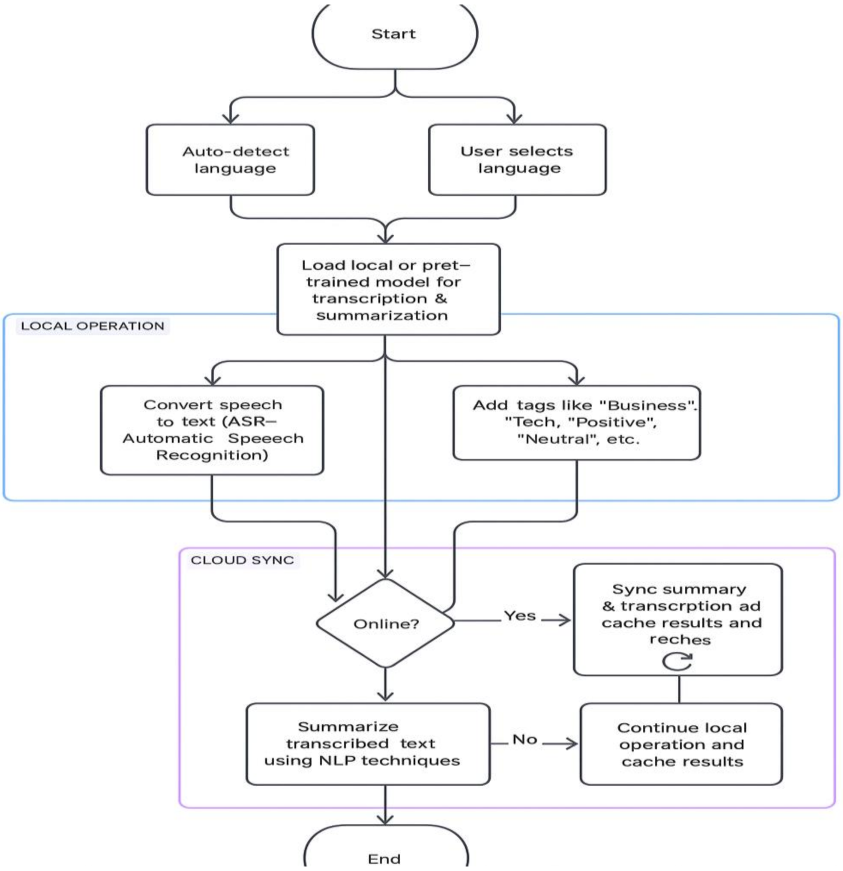


Literature Survey

Author	Methodology	Features	Challenges
Raffel, C., et al. (2020)	Developed T5, a text-to-text transformer model for NLP tasks, including summarization.	Achieves state-of-the-art text summarization with transfer learning.	Requires high computational resources and extensive training data.
Zhang, Y., et al. (2021)	Applied BERT-based extractive summarization for podcast transcripts.	Extracts key sentences while maintaining contextual meaning.	Computationally expensive for long audio content.

Luo, Y., et al. (2022)	Developed Whisper, an ASR model optimized for multilingual speech transcription.	Provides robust transcription across various languages and noisy environments.	Requires large-scale datasets and fine-tuning for specific domains.
Kong et al. (2022)	Proposed an end-to-end pipeline combining ASR and transformer-based summarization for podcast content	Demonstrated modular designs can outperform monolithic summarizers by customizing each step.	Emphasized the need for chunking, context retention, and speaker
PodcastSum – Yang et al. (2021)	Introduced a benchmark dataset for podcast summarization with aligned transcripts and human-written summaries.	Facilitated evaluation of summarization algorithms on informal, spoken data.	Dataset is limited in domain variety and episode length.

Flow Chart


Start
Auto-detect language
User selects language
Load local or pret–trained model for transcription & summarization
LOCAL OPERATION
Convert speech to text (ASR–Automatic Speeech Recognition)
Add tags like "Business". "Tech, "Positive", "Neutral", etc.
CLOUD SYNC
Online?
Yes
Sync summary & transcrption ad cache results and reches
Summarize transcribed text using NLP techniques
No
Continue local operation and cache results
End

The flowchart starts with the user either uploading a podcast or recording one directly. This is the starting point for the AI to begin working.

Next, the system checks the language of the podcast. It can either detect the language automatically or let the user choose it manually. Once the language is set, the AI loads a model to process the podcast. This model could be stored locally or already pre-trained to handle transcription and summarization. In the local operation part, the AI first converts the speech into text using speech-to-text technology. After that, it can also add some helpful tags like “Tech,” “Business,” or “Positive,” based on the podcast content. Then comes the internet check. If the device is online, it will sync the transcription and summary to the cloud for backup and further use. If the device is offline, it will continue working locally and store the results on the device. Next, the AI uses natural language processing to summarize the transcribed text. This turns the long podcast into a short, easy-to-read version.

Finally, the system shows the user the summary, the full transcription, and any tags. And that’s the end of the flow.

Proposed Methodology

The proposed methodology aims to automate the summarization of podcast content using AI technologies such as speech recognition, natural language processing (NLP), and cloud integration. The system follows a modular, step-by-step approach to ensure efficient processing, accuracy, and user-friendliness.

To build an effective AI-based podcast summarizer, we’ve broken down the process into a set of simple and logical steps. The main goal is to take podcast audio, understand what’s being said, and turn it into a neat, readable summary that saves users time. Here's how the system is designed to work:

1. Getting the Podcast Audio

The first step is to collect the podcast audio. Users can either upload an existing podcast file (like an MP3) or record one directly through the platform. This gives flexibility to content creators and listeners alike.

2. Detecting the Language

Once the audio is ready, the system checks which language is being spoken. It can automatically detect it, or the user can choose the language manually. This helps ensure that the transcription and summarization models work accurately.

3. Preparing the System

After knowing the language, the system loads the required models for transcription and summarization. If the device has enough power, it can run these models locally. If not, or if better performance is needed, it can connect to the cloud to use more advanced models.

4. Converting Speech to Text

Now comes the important part—converting the spoken words into written text. This is done using speech-to-text (ASR) technology. The output is a complete transcript of the podcast, including all the spoken content.

5. Summarizing the Content

Once we have the transcript, the AI analyzes the text using Natural Language Processing (NLP). It identifies key points, removes unnecessary fluff, and creates a short and meaningful summary that captures the essence of the episode.

6. Adding Extra Insights

If needed, the system can go a step further by tagging the podcast with topics (like “Technology,” “Finance,” etc.) and detecting the overall tone or sentiment (positive, negative, neutral). These tags help in organizing and recommending podcasts more efficiently.

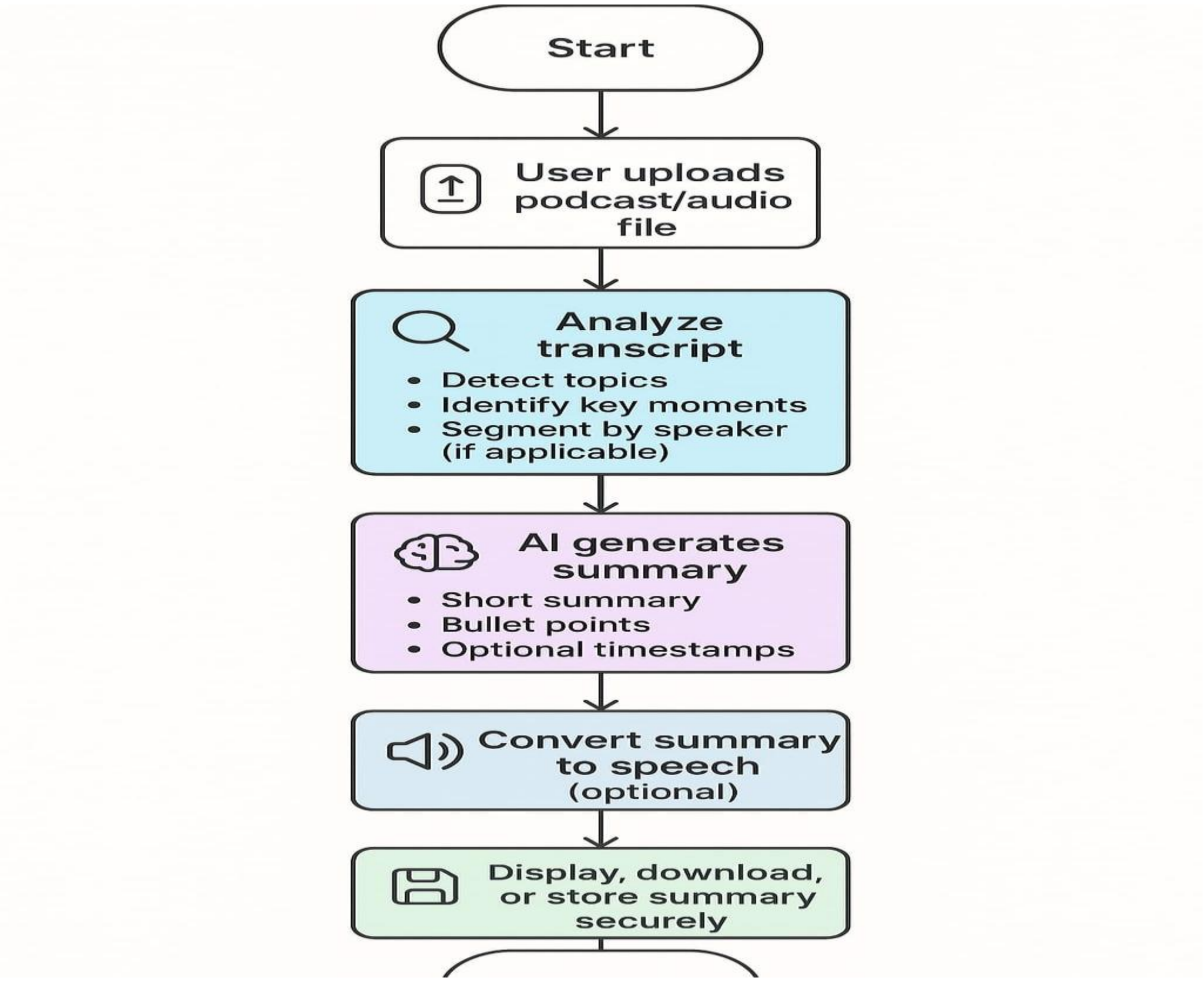
7. Cloud Sync

If the system is connected to the internet, the transcript and summary are saved to the cloud. This allows users to access their results from other devices or share them. If the system is offline, everything is saved locally and synced later when the internet is available.

8. Showing the Final Result

At the end, the user gets a clean, user-friendly output: the summarized version of the podcast, the full transcript, and any tags or extra analysis. This allows users to quickly understand the content without listening to the whole episode.

Work Flow Chart


Start
User uploads podcast/audio file
Analyze transcript
• Detect topics
• Identify key moments
• Segment by speaker (if applicable)
AI generates summary
• Short summary
• Bullet points
• Optional timestamps
Convert summary to speech (optional)
Display, download, or store summary securely

References

[1] A. Radford et al., "Whisper: Robust Speech Recognition via Large-Scale Weak Supervision," OpenAI, 2022. [Online]. Available: https://openai.com/research/whisper

[2] T. Wu et al., "WhisperX: Fast and Accurate Automatic Speech Recognition with Wordlevel Alignment," GitHub Repository, 2023. [Online]. Available: https://github.com/mbain/whisperx

[3] J. Zhang et al., "PEGASUS-X: Scaling PEGASUS with Mixture-of-Experts for Long Document Summarization," arXiv preprint arXiv:2206.04416, 2022.

[4] S. Guo et al., "LongT5: Efficient Text-To-Text Transformer for Long Sequences," arXiv preprint arXiv:2112.07916, 2022.

[5] OpenAI, "GPT-4 Technical Report," arXiv preprint arXiv:2303.08774, 2023.

[6] K. Yang et al., "PodcastSum: A Dataset for Podcast Summarization," arXiv preprint arXiv:2101.02409, 2021.

[7] H. Aliannejadi et al., "ClariQ: A Large-Scale Dataset for Clarification in InformationSeeking Conversations," in Proc. of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2021, pp. 1937–1946.

[8] Q. Kong, Y. Xu, and W. Wang, "A Modular Framework for Podcast Summarization Using Transformer-Based Models," in Proc. of IEEE Spoken Language Technology Workshop (SLT), 2022.

[9] A. Turan et al., "Speech-to-Summary: Real-Time Transcription and Summarization Pipeline for Audio Media," in Proc. of International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022

[10] M. Lewis et al., *"BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension,"*arXiv preprint arXiv:1910.13461, 2019.

[11]A.Baevski et al., *"wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations,"*arXiv preprint arXiv:2006.11477, 2020.

[12]J. Devlin et al., *"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,"*arXiv preprint arXiv:1810.04805, 2018.