

AI-Based Lecture Notes Generation System Using Speech Recognition and Natural Language Processing

Athmika PA ¹, Harini N ², Jananitharmi B ³, Nevethe SK⁴, and Mrs. Padma Priya R ⁵

1,2,3,4 UG Students, 5 Assistant Professor, Department of Computer Science and Engineering,
School of Engineering, Avinashilingam Institute for Home Science and Higher Education for Women,
Coimbatore 641108, Tamil Nadu, India

Abstract

The rapid growth of recorded educational content has increased the amount of lecture material that students must review and organize. Manual note-taking during fast-paced lectures may cause important concepts to be missed and requires additional time for revision. This paper presents an AI-based lecture notes generation system that converts uploaded or recorded lecture audio into structured digital learning material. The proposed system accepts lecture audio through a web interface and uses a Flask-based backend to coordinate audio preprocessing, speech recognition, and natural language processing. The audio is prepared using Librosa and noise-reduction techniques before OpenAI Whisper is used to generate a textual transcript. The transcript is subsequently processed using NLP techniques, with RAKE used to extract important keywords and a transformer-based T5-family model used for summarization and question-answer generation. The resulting transcript, keywords, summary, structured notes, and question-answer

content are presented to the user through the application interface. The project integrates established speech and language technologies into a single educational workflow and is intended to reduce manual note-preparation effort. No numerical accuracy, dataset size, processing-time, or user-study results are reported in this paper because validated experimental measurements were not supplied with the project materials.

KeyWords — *Lecture Notes Generation, Automatic Speech Recognition, Whisper, Natural Language Processing, Text Summarization, Keyword Extraction, Question Answering.*

I. INTRODUCTION

Recorded lectures, online classes, and digital learning platforms have made large amounts of spoken educational content available to students. Although recorded material allows learners to revisit a lecture, reviewing a long recording repeatedly can be time-consuming. Students also commonly attempt to create notes while listening, which requires simultaneous attention to speech, comprehension, selection of important information, and writing.

Traditional note-taking may therefore produce incomplete or inconsistent notes, particularly when a lecturer presents information rapidly or when the student is distracted. A raw audio recording is also difficult to search, while a complete transcript may contain considerably more information than a student needs during revision. An automated method that converts speech into concise and organized learning material can reduce these difficulties.

Automatic speech recognition (ASR) provides the first step by converting spoken language into text. Deep neural approaches have significantly influenced modern ASR, and end-to-end speech recognition has become an important research direction. Whisper further extends this direction through large-scale weakly supervised training and multilingual, multitask speech processing.

Once speech has been transcribed, natural language processing can be used to identify important concepts and create shorter representations of the content. Keyword extraction methods such as RAKE provide an unsupervised way of identifying important phrases, while transformer-based text-to-text models such as T5 support summarization and question answering.

The proposed project combines these capabilities into one educational application. Its main contribution is the integration of audio preprocessing, Whisper transcription, NLP-based transcript processing, RAKE

keyword extraction, T5-based summarization, structured note generation, and question answering within a single lecture-processing workflow. The project is implemented using Python, Flask, HTML, CSS, JavaScript, Librosa, Noisereduce, Transformers, T5, and RAKE-NLTK.

II. LITERATURE SURVEY

Research in speech recognition has progressed from statistical and hybrid acoustic models toward end-to-end neural systems. Hinton et al. described the use of deep neural networks for acoustic modeling and reported the advantages of deep architectures over earlier Gaussian mixture model approaches in speech recognition. Prabhavalkar et al. later surveyed end-to-end ASR architectures and their development, training, decoding, and deployment characteristics. These developments provide the broader technical foundation for converting lecture audio into text.

Whisper, introduced by Radford et al., uses large-scale weak supervision and demonstrates robust zero-shot speech processing across multilingual and multitask settings. Its pretrained nature makes it suitable for an application in which the project does not train a speech-recognition model from scratch. The proposed system uses Whisper as the speech-to-text stage after audio preprocessing.

Keyword extraction is another important stage in converting long text into compact learning material. Rose et al. introduced Rapid Automatic Keyword Extraction (RAKE), an unsupervised, domain-independent method for extracting keyword phrases from individual documents. Graph-based approaches such as TextRank have also been applied to keyword and sentence ranking. The proposed system selects RAKE because it provides a lightweight document-level extraction mechanism.

For summarization and language generation, transformer-based encoder-decoder models have become widely used. Raffel et al. proposed T5 as a unified text-to-text transfer-learning framework that can address multiple NLP tasks, including summarization and question answering. Zhang et al. introduced PEGASUS with a pretraining objective designed for abstractive summarization. These studies support the use of transformer-based generation for reducing lengthy lecture transcripts.

Question answering has also developed through transformer-based language understanding. Devlin et al. introduced BERT and demonstrated strong performance on several language-understanding tasks,

including question answering. Rajpurkar et al. introduced the SQuAD dataset as a benchmark for machine comprehension and extractive question answering. The proposed project uses a T5-family generative approach rather than a BERT-style extractive QA pipeline.

For summary evaluation, ROUGE is a commonly used automatic metric for comparing generated summaries with reference summaries. However, the present project does not contain validated ROUGE measurements. Similarly, speech-recognition evaluation should use an established measure such as Word Error Rate (WER), but the project materials do not provide a test dataset or WER results.

III. PROBLEM STATEMENT

Students often find it difficult to create complete and organized notes while simultaneously listening to a lecture. Important explanations, definitions, examples, and keywords can be missed when the lecture progresses quickly. Although audio recordings can be replayed, repeated listening increases revision time and does not directly provide concise study material.

Existing tools may provide transcription or summarization as separate functions, but a student-oriented workflow benefits from connecting speech recognition with keyword extraction, summarization, note organization, and question answering. A raw transcript is also not equivalent to structured notes because it may contain repetitions, filler expressions, and information that is less important for revision.

The problem addressed by this project is therefore the development of an integrated AI-based system that accepts lecture audio and automatically transforms it into a textual transcript, important keywords, concise summary, structured lecture notes, and content-based questions and answers. The system should provide these functions through an accessible web interface while keeping the processing workflow modular.

IV. EXISTING SYSTEM

Existing lecture-support approaches can be broadly grouped into manual note-taking, standalone speech transcription, standalone summarization, and general-purpose educational or meeting assistants. Manual note-taking requires the learner to listen and write at the same time. This can be effective for short lectures but becomes difficult when the volume and speed of information increase.

Standalone ASR tools can convert speech into text, providing a searchable representation of a lecture.

However, transcription alone may produce a long document that still requires manual processing. Students may have to read the complete transcript to identify the important concepts and create their own summary.

Text summarization systems reduce the amount of information that must be read, but they generally require text as input. Consequently, when the original material is a recorded lecture, an additional speech-to-text stage is necessary before summarization.

Keyword extraction systems can identify important terms, while question-answering systems can support interactive learning. However, when these functions are provided separately, the student must move information between different tools. The proposed system combines these operations so that the output of the speech-recognition stage directly feeds the NLP stages.

V. LIMITATIONS OF EXISTING SYSTEM

A major limitation of manual lecture-note preparation is the time required to listen, understand, and write simultaneously. The method also depends strongly on individual note-taking skill and attention. When the lecture is fast-paced, students may omit details or write incomplete statements.

Standalone transcription tools reduce the writing burden but can leave the learner with a lengthy transcript. Speech-recognition quality can also be affected by background noise, accents, overlapping speech, recording quality, and domain-specific terminology.

Standalone summarization systems require a textual source and therefore do not directly solve the audio-to-notes problem. Automated summaries may also omit important information or produce wording that requires verification against the original lecture.

Separate keyword and question-answering tools create additional interaction steps. They may also lack awareness of the lecture context unless the user manually supplies the appropriate text. These limitations motivate an integrated workflow that processes lecture audio from input to structured learning output.

VI. PROPOSED SYSTEM

The proposed AI-based Lecture Notes Generation System is a web application that transforms lecture audio into structured digital learning material. The system follows a modular pipeline in which each stage

performs a specific processing function and passes its output to the next stage.

A. User Interface and Audio Input

The user interface is developed using HTML, CSS, and JavaScript. It provides an entry point for uploading lecture audio and displaying generated results. The project materials specify support for common audio formats such as MP3, WAV, and MPEG. The input module can receive a previously recorded lecture and pass it to the backend.

B. Audio Preprocessing and Noise Reduction

The audio preprocessing module prepares the input for speech recognition. Librosa is used for loading and analyzing audio signals, while Noisereduce is used in the project to reduce unwanted background noise. Where required, audio conversion and chunking can be performed before transcription.

C. Whisper Speech Recognition

The preprocessed audio is supplied to OpenAI Whisper. Whisper converts the spoken lecture into a text transcript. The transcript is the principal intermediate representation used by the downstream NLP modules. The project does not train Whisper from scratch; it uses the pretrained speech-recognition capability.

D. Transcript Processing

The transcript is cleaned and prepared for language processing. Text preprocessing may include normalization, removal of unwanted content, stop-word handling, and sentence organization. The project materials identify NLTK-based processing for text cleaning.

E. Keyword Extraction

RAKE identifies important terms and phrases from the processed transcript. The method uses candidate phrases separated by stop words and punctuation and ranks candidate keywords according to word-level statistics. The resulting keyword list provides a compact representation of major lecture concepts.

F. Text Summarization

A transformer-based T5-family model is used to generate a concise summary of the processed lecture text. T5 formulates language tasks in a text-to-text framework. The project description supplied for this paper specifies T5-base, while the project presentation lists FLAN-T5-Small for summarization and question generation. The exact deployed checkpoint must therefore be verified before final publication.

G. Structured Note Generation

The transcript, summary, and extracted keywords are organized into readable lecture notes. The purpose is not to replace the original lecture but to provide a concise revision-oriented representation of the information.

H. Question Answering

The question-answering facility enables the user to ask questions based on the generated lecture content. The language model generates answers using the available lecture context. The exact context construction, prompt, and decoding parameters should be reported from the final implementation.

TABLE I. TECHNOLOGIES USED IN THE PROPOSED SYSTEM

Component	Technology / Model	Purpose
Frontend	HTML, CSS, JavaScript	User interface
Backend	Flask / Python	Workflow management
Audio processing	Librosa	Audio processing
Noise reduction	Noisereducer	Noise removal
Speech recognition	OpenAI Whisper	Speech-to-text
Keyword extraction	RAKE-NLTK	Keyword extraction
Summarization	T5-family model	Text summary
NLP processing	NLTK	Text processing
Question answering	T5-family model	Lecture-content Q&A

VII. SYSTEM ARCHITECTURE

The system architecture connects the user interface, audio-processing components, speech-recognition module, NLP modules, and output interface. The complete logical flow is: Lecture Audio → Audio Upload/Recording → Audio Preprocessing → Noise Reduction → Whisper Speech Recognition → Text Transcript → NLP Processing → Keyword Extraction + Summarization → Structured Lecture Notes → Question Answering.

The Flask backend receives the uploaded audio and controls the processing sequence. After preprocessing, the audio is passed to Whisper, which produces the transcript. The transcript is then cleaned and supplied to RAKE and the summarization model. The outputs are

combined into a structured representation and made available through the web interface.

The architecture is modular, allowing individual components to be replaced or improved without redesigning the complete application. For example, a different ASR model can be evaluated without changing the keyword extraction interface, while a stronger summarization model can replace the current T5-family component.

The audio preprocessing workflow includes loading the recording, applying noise reduction, converting the signal when required, and preparing it for speech recognition. The output is a cleaned audio representation suitable for Whisper.

The speech-to-text stage converts the prepared lecture audio into a transcript. The transcript is subsequently passed to the NLP stage for cleaning, keyword extraction, summarization, and question-answer processing.

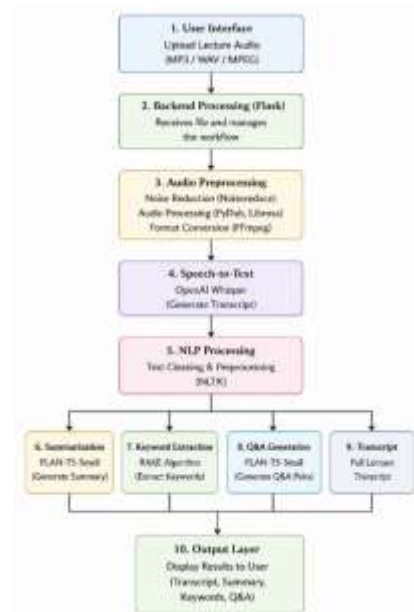


Fig. 1. Architectural design.

VII. OUTPUT AND RESULT

The prototype is intended to produce multiple outputs from one lecture recording: a speech transcript, extracted keywords, a concise summary, structured lecture notes, and question-answer content. The supplied project presentation includes an output section and describes the system as generating organized lecture content. However, it does not provide a reproducible numerical evaluation, and therefore this section does not claim specific accuracy or performance values.

A. Speech-to-Text Output

The transcript should contain the recognized lecture speech in textual form. A publication-ready evaluation should compare the generated transcript with a manually verified reference transcript and report Word Error Rate (WER). Required result:

B. Generated Summary

The summarization stage should reduce the length of the transcript while retaining the principal lecture concepts.

C. Extracted Keywords

The keyword output should represent the central concepts discussed in the lecture. If a manually prepared reference keyword list is available, precision, recall, and F1-score can be calculated.

D. Generated Lecture Notes

The final notes should combine the summary and keywords with the relevant transcript information in a readable structure. The recommended output figure should show the actual interface and the generated note content.

E. Question-Answering Output

The question-answering interface should allow a user to submit a lecture-related question and receive an answer grounded in the processed content.

F. Overall System Performance

The overall system should be evaluated using actual measurements rather than qualitative assumptions. Recommended measurements include processing time per audio file, memory or GPU requirements, transcription quality, summarization quality, keyword quality, question-answer quality, and user evaluation.

The project materials do not provide dataset size, participant count, accuracy values, processing time, or user-study scores. These values are therefore intentionally left as placeholders and must be replaced only after the corresponding experiments are performed.

```
API POST /process - File upload and processing started
[INFO] request files kept: ['file']
[INFO] request form keys: ['summary', 'qqa']
[INFO] request files: sampleaudio.mp3, filestorage: 'sampleaudio.mp3' ('audio/1732')
[INFO] File object received: filestorage: 'sampleaudio.mp3' ('audio/1732')
[INFO] File saved to: uploads/sampleaudio.mp3
[INFO] File size: 20000 bytes
[INFO] Generate Summary: True, Generate Q&A: True
[INFO] File is mp3, converting to wav...
[INFO] Converting sampleaudio.mp3 to wav...
[INFO] Successfully converted to uploads/sampleaudio.wav
[INFO] Applying noise reduction...
[INFO] Resizing video for uploads/sampleaudio.wav...
[INFO] Noise reduction complete: processed.wav
[INFO] Starting transcription...
[INFO] Transcription complete. Text length: 2297 characters
[INFO] Generating summary...
[INFO] Summary generated. Length: 393 characters
[INFO] Generating Q&A...
[INFO] Q&A generated. Length: 488 characters
[INFO] Removing temporary audio files...
[INFO] Removed: uploads/sampleaudio.mp3
[INFO] Removed: uploads/sampleaudio.wav
[INFO] Removing generated files...
[INFO] Removing generated files...
```

Fig. 2. Text Processing.

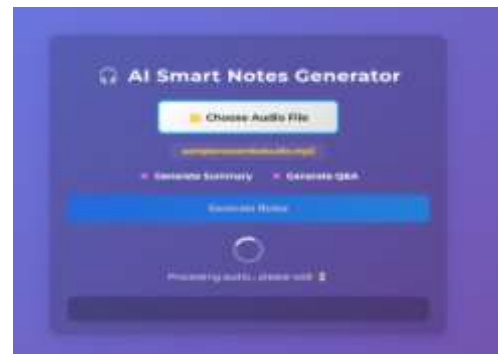


Fig. 3. Choosing of audio file..

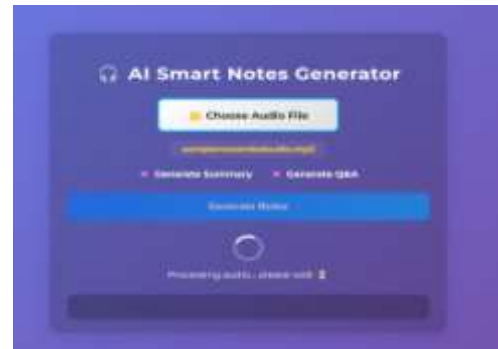


Fig. 4. Notes generation through the audio file.

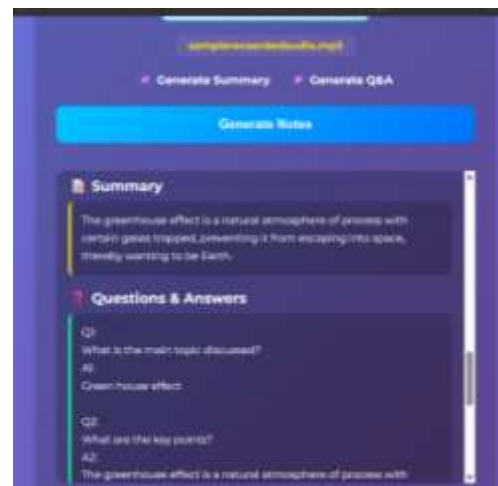


Fig. 5. Output summary and Q&A generation

IX. CONCLUSION AND FUTURE SCOPE

The AI-based Lecture Notes Generation System provides an integrated approach for converting recorded lecture audio into structured digital learning material. The system combines audio preprocessing, Whisper-based speech recognition, NLP-based transcript processing, RAKE keyword extraction, transformer-based summarization, structured note generation, and question answering. The modular Flask-based backend coordinates the processing stages, while HTML, CSS, and JavaScript provide the user-facing interface.

The main usefulness of the proposed system is its ability to connect speech recognition with downstream language-processing functions rather than stopping at

transcription. The resulting representation can support revision by presenting a transcript, important keywords, a concise summary, structured notes, and content-related questions and answers. The system therefore provides a practical foundation for AI-assisted educational content processing.

The current work has several limitations. Speech-recognition quality may vary with noise, accents, overlapping speech, and recording conditions. Long recordings can require substantial processing time. Abstractive summaries may omit details or require verification against the source transcript. In addition, the exact T5-family checkpoint used in the final implementation must be verified because the project description and project presentation identify different model variants.

Future development can extend the system toward multilingual speech support, real-time lecture transcription, speaker identification, improved summarization, retrieval-grounded question answering, cloud deployment, mobile applications, and personalized learning features. Future experiments should also establish a representative evaluation dataset and report WER, summarization metrics, keyword-extraction metrics, QA performance, processing time, and user evaluation.

X. REFERENCES

- [1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury, and T. N. Sainath, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012, doi: 10.1109/MSP.2012.2205597.
- [2] R. Prabhavalkar, T. Hori, T. N. Sainath, R. Schlüter, and S. Watanabe, "End-to-end speech recognition: A survey," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 32, pp. 325–351, 2024, doi: 10.1109/TASLP.2023.3328283.
- [3] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv:2212.04356*, 2022.
- [4] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," in *Text Mining: Applications and Theory*. Chichester, U.K.: Wiley, 2010, pp. 1–20, doi: 10.1002/9780470689646.ch1.
- [5] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.
- [6] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proc. 2004 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Barcelona, Spain, 2004, pp. 404–411.
- [7] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," in *Proc. 37th Int. Conf. Machine Learning (ICML)*, 2020, pp. 11328–11339.
- [8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. 2019 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, MN, USA, 2019, pp. 4171–4186.
- [9] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proc. 2016 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Austin, TX, USA, 2016, pp. 2383–2392, doi: 10.18653/v1/D16-1264.
- [10] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain, 2004, pp. 74–81.
- [11] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in Python," in *Proc. 14th Python in Science Conf. (SciPy)*, 2015, pp. 18–25, doi: 10.25080/Majora-7b98e3ed-003.
- [12] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Machine Learning (ICML)*, 2006, pp. 369–376, doi: 10.1145/1143844.1143891.