IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

AI Lecture-to-Notes Generator

Nitin Shravan Yeole[1], Manoj V. Nikum[2], Shivam B. Limbhare[3]

MCA Student, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India[1]

HOD MCA & Assistant Professor, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India[2]

Assistant Professor, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India [3]

Abstract: Students lose a great deal of attention during lectures because listening, understanding and writing must happen at the same time, and revising from long lecture recordings is slow. This paper presents the design of an AI Lecture-to-Notes Generator that converts lecture audio or video into structured, revisable study notes. The system uses OpenAI Whisper, a transformer-based speech recognition model, to transcribe the lecture. The raw transcript is cleaned by removing filler words and repeated phrases and is divided into topic-wise chunks that fit the input limit of the summarisation model. Each chunk is passed to BART, a transformer sequence-to-sequence model, which produces a concise abstractive summary. The summaries are merged and formatted into headings, bullet points and key terms, and the final notes can be downloaded as a PDF or Word file through a simple web interface. Transcription quality is measured with Word Error Rate (WER) and summary quality with ROUGE scores, together with a comparison against manually written notes. The paper documents the architecture, processing parameters and evaluation protocol, and reviews the published benchmark performance of the two underlying models. The proposed tool aims to save students' time, support better revision and help learners who find real-time note-taking difficult.

Keywords: Lecture to notes, Automatic speech recognition, Whisper, BART, Abstractive text summarisation, Natural language processing, ROUGE, Word Error Rate

I. INTRODUCTION

A lecture is a continuous stream of spoken information. A student sitting in a classroom must listen to the teacher, understand the idea being explained, decide what is important and write it down, all within a few seconds before the next idea arrives. Research on note-taking describes it as a cognitively demanding activity in which the writing task competes with listening and comprehension for limited working-memory capacity [1]. As a result, students often miss key points, produce incomplete notes or copy sentences without understanding them. Mueller and Oppenheimer found that learners who transcribe lectures almost word for word tend to perform worse on conceptual questions than learners who summarise the content in their own words [2]. Good notes are therefore not a copy of the lecture; they are a shorter, organised and reworded version of it.

Many institutions now record lectures, and online courses are delivered mainly as video. Recordings solve the problem of missing a lecture, but they create a new one. A one-hour lecture needs an hour to replay, it cannot be scanned like a page of text, and finding one definition or formula inside the recording requires repeated seeking. During examination time, students do not have the time to re-watch every lecture. They need the content in a compact written form that can be read in minutes and revised repeatedly.

Two recent developments in deep learning make it practical to automate this conversion. The first is robust speech recognition. Whisper, trained on 680,000 hours of multilingual and multitask supervised data collected from the web, approaches human-level accuracy on many English benchmarks without any task-specific fine-tuning [3]. The second is pretrained sequence-to-sequence language models. BART combines a bidirectional encoder with an autoregressive decoder and reaches state-of-the-art results on abstractive summarisation tasks [4]. Both models are openly available through the Hugging Face Transformers library [5], which makes it possible for a student project to combine them without training a model from scratch.

This paper proposes and describes an AI Lecture-to-Notes Generator that joins these two models into one pipeline. A user uploads a lecture recording through a web page. The system extracts the audio, transcribes it with Whisper, cleans the transcript, splits it into topic-wise chunks, summarises each chunk with BART and arranges the summaries into headings, bullet points and key terms. The final notes can be downloaded as a PDF or Word file.

The main contributions of this work are as follows. (1) An end-to-end architecture that converts lecture audio or video into structured study notes using only openly available pre-trained models. (2) A chunking strategy that respects the 1024-token input limit of BART while keeping each chunk topically coherent. (3) A note-structuring step that turns unformatted summaries into headings, bullet points and key terms. (4) An evaluation protocol that combines Word Error Rate, ROUGE scores and a comparison with manually written notes.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

The remainder of this paper is organised as follows. Section II reviews related work on note-taking support, speech recognition and text summarisation. Section III describes the proposed methodology. Section IV presents the experimental configuration, the published benchmark results of the underlying models and the evaluation results format. Section V concludes the paper and outlines future work.

II. LITERATURE REVIEW

A. Note-Taking and Lecture Capture

Studies of note-taking show that the value of notes depends on how much processing the student performs while writing them [1], [2]. Verbatim transcription is quick for typists but encourages shallow processing, while selective, reworded notes support understanding. This finding motivates abstractive rather than purely extractive summarisation in an automatic note generator: notes that restate the content concisely are closer to what an attentive student would write.

B. Automatic Speech Recognition

Early large-vocabulary speech recognisers combined hidden Markov models with Gaussian mixture acoustic models. Deep neural networks later replaced the Gaussian mixtures and produced large reductions in word error rate [6]. Connectionist Temporal Classification removed the need for frame-level alignments and made end-to-end training possible [7]. Self-supervised learning pushed accuracy further: wav2vec 2.0 learns speech representations from unlabelled audio and, after fine-tuning on a small labelled set, reaches a WER of 1.8% on the LibriSpeech test-clean set when decoded with a language model [8], [9]. A weakness of such systems is that a model fine-tuned on one corpus can degrade when accents, microphones or vocabulary change.

Whisper takes a different route. It is a standard encoder-decoder transformer trained on a very large and diverse collection of weakly supervised audio-transcript pairs rather than on a single clean corpus [3]. Input audio is resampled to 16 kHz, converted to an 80-channel log-Mel spectrogram and processed in 30-second windows. Five model sizes are released, from 39 million to 1.55 billion parameters, so that users can trade accuracy for speed. Although zero-shot Whisper does not beat models fine-tuned on LibriSpeech on that specific benchmark, it makes substantially fewer errors on average across a wide range of other test sets, which indicates better robustness to the conditions found in real classrooms [3]. This robustness is the main reason for choosing Whisper for lecture audio.

C. Text Summarisation

Automatic summarisation has been studied since the 1950s. The earliest systems scored sentences by word frequency and selected the top-ranked ones [10]. Graph-based methods such as TextRank rank sentences by their centrality in a similarity graph [11]. Surveys of the field describe extractive methods, which copy sentences from the source, and abstractive methods, which generate new sentences [12]. Extractive output is grammatical but often redundant, and spoken language makes the problem harder because the best sentences in a lecture are rarely self-contained.

The transformer architecture [13] and large-scale pre-training changed the field. BERT-based encoders were adapted for both extractive and abstractive summarisation [14], [15]. Text-to-Text Transfer Transformer (T5) treats every task as text generation [16], and PEGASUS uses a pre-training objective designed for summarisation, in which whole sentences are masked and must be generated [17]. BART is pre-trained as a denoising autoencoder: text is corrupted with an arbitrary noising function, and the model learns to reconstruct the original [4]. After fine-tuning on the CNN/DailyMail corpus [18], BART-large obtained ROUGE-1, ROUGE-2 and ROUGE-L scores of 44.16, 21.28 and 40.90 respectively, competitive with the best contemporary systems [4]. Its encoder accepts at most 1024 tokens, so documents as long as a lecture transcript must be divided before summarisation. Long-input transformers such as Longformer reduce this limit with sparse attention [19] but they require additional models and memory.

D. Evaluation Metrics

Speech recognition is evaluated with Word Error Rate, which counts substitutions, deletions and insertions against a reference transcript [20]. Summaries are evaluated with ROUGE, which measures n-gram and longest-common-subsequence overlap between a system summary and one or more reference summaries [21]. ROUGE correlates reasonably with human judgement at system level but cannot judge factual accuracy, so human comparison remains necessary.

E. Research Gap

Most published work evaluates speech recognition and summarisation separately, on read speech or news text. Lecture speech is different from both: it is spontaneous, contains hesitations and repetitions, mixes everyday language with technical terms and runs for a long time. Summarisation models trained on news articles, such as BART fine-tuned on CNN/DailyMail, are also not trained on spoken lectures. Dialogue-style corpora such as SAMSum illustrate that spoken, conversational text needs its own treatment [22]. A practical tool for students must therefore connect the two models,

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

handle the long-input limit, clean spoken-language noise, and present the output in a form that students can use directly. This paper addresses that gap through a modular pipeline and an evaluation that measures both stages.

III. METHODOLOGY

A. System Overview

The proposed system is a sequential pipeline of seven stages, shown in Fig. 1: (i) upload of the recording, (ii) audio extraction, (iii) speech recognition, (iv) transcript cleaning, (v) topic-wise chunking, (vi) abstractive summarisation and (vii) note structuring and export. Each stage has a single responsibility and a plain-text interface, so a stage can be replaced (for example, a larger Whisper model) without changing the others.

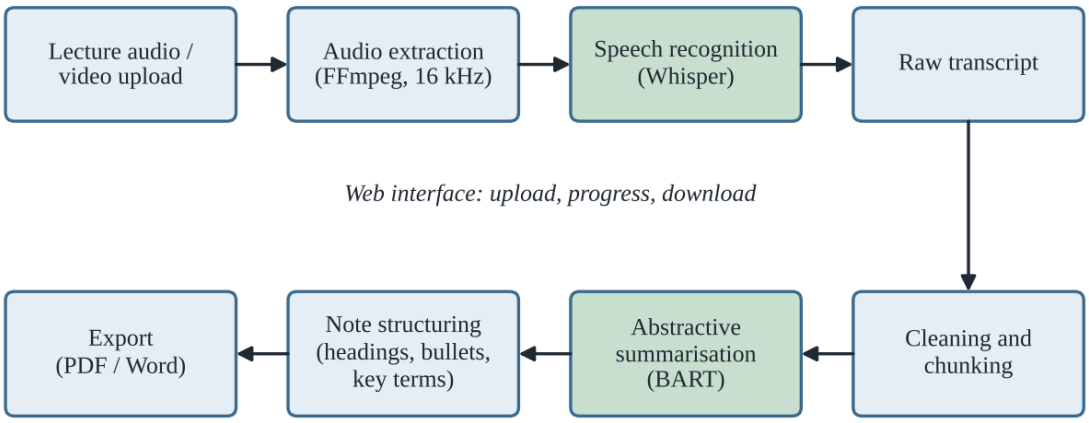


Fig. 1 Architecture of the proposed AI Lecture-to-Notes Generator

B. Audio Acquisition and Transcription

The user uploads an audio file (for example MP3, WAV, M4A) or a video file (for example MP4). FFmpeg extracts the audio track and converts it to a 16 kHz mono waveform, which is the input format expected by Whisper [3]. Whisper then performs transcription. The decoder produces text tokens conditioned on the encoder's representation of each 30-second window, and the long-form transcription procedure slides the window over the recording to cover the whole lecture. Five Whisper sizes are available (Table I), and the system allows the user to select one, with the small or base model as a practical default on ordinary hardware. The language is detected automatically but can be fixed by the user to avoid detection errors on short or noisy recordings.

TABLE I WHISPER MODEL SIZES [3]

Model	Parameters	Approx. memory needed
Tiny	39 M	~1 GB
Base	74 M	~1 GB
small	244 M	~2 GB
medium	769 M	~5 GB
large	1550 M	~10 GB

The memory column is the approximate GPU memory requirement published with the open-source release of Whisper. Larger models are more accurate but slower, so the choice depends on the available hardware and the accuracy required.

C. Transcript Cleaning

Raw spoken-language transcripts contain features that do not belong in written notes. The cleaning stage applies the following rule-based steps: (1) removal of filler words and hesitation sounds such as “um”, “uh”, “you know” and “I mean”; (2) collapse of immediately repeated words and phrases (“we will we will see” becomes “we will see”); (3) normalisation of whitespace and punctuation; (4) capitalisation of the first word of each sentence; and (5) sentence segmentation. Cleaning is intentionally conservative: it removes only material that carries no information, because deleting content words would damage the summary. Technical terms and numbers are never altered.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

D. Topic-Wise Chunking

BART accepts at most 1024 input tokens [4], while a one-hour lecture can contain more than 8,000 words. The cleaned transcript is therefore divided into chunks. A simple fixed-length split would cut sentences and ideas in half, so the proposed procedure works at sentence level. Sentences are grouped sequentially into a chunk until adding the next sentence would exceed a token budget of 900 tokens, which leaves a margin for special tokens. Before a chunk is closed, the procedure looks for a topic boundary near the limit using lexical cohesion, in the manner of TextTiling [23]: the vocabulary overlap between the sentences before and after each candidate position is measured, and the position with the lowest overlap is chosen as the break. A short overlap of one sentence is carried into the next chunk to preserve context. Fig. 2 illustrates the resulting chunk-and-summarise flow.

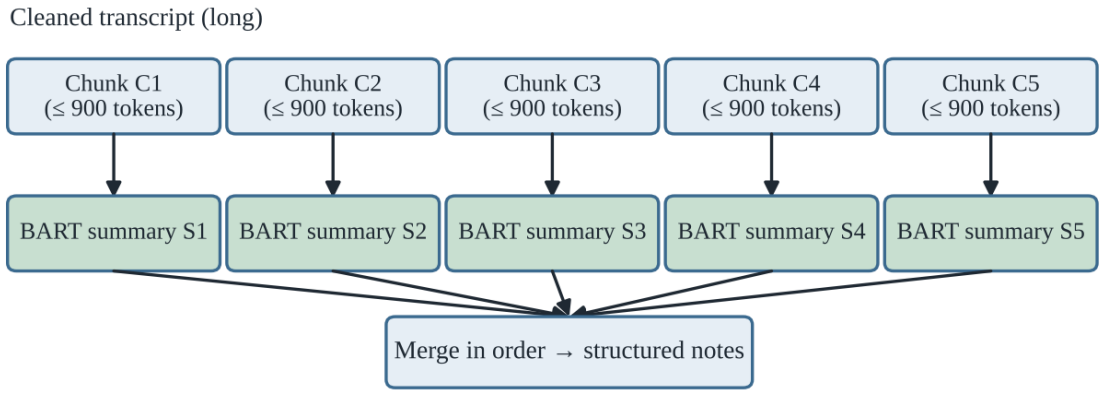


Fig. 2 Chunking and chunk-wise abstractive summarisation

E. Abstractive Summarisation with BART

Each chunk is summarised independently by BART-large fine-tuned on CNN/DailyMail (the publicly released facebook/bart-large-cnn checkpoint). The encoder reads the chunk bidirectionally and the decoder generates the summary one token at a time while attending to the encoder output [4]. Generation uses beam search with four beams, a length penalty of 2.0, a minimum length of 56 and a maximum length of 142 tokens (the defaults of the released checkpoint), and a three-gram repetition block to avoid repeated phrases. For chunks that are shorter than the minimum summary length, the length limits are scaled down in proportion to the chunk length so that the model does not pad a short chunk with invented content. Chunk summaries are kept in lecture order so that the notes follow the same sequence as the lecture.

The summaries produced by a news-trained checkpoint may reflect the style of news writing. The pipeline is designed so that the checkpoint can be replaced with a version fine-tuned on lecture or conversational data without other code changes, which is discussed under future work.

F. Note Structuring and Key-Term Extraction

Chunk summaries are plain paragraphs, so a final structuring stage converts them into study notes. First, each summary is split into sentences and each sentence becomes one bullet point. Second, a heading is produced for every chunk from its highest-weighted key terms, so that students can scan the notes by topic. Third, key terms are extracted from the whole transcript using term frequency–inverse document frequency (TF-IDF) computed over the chunks, with stop words removed, and the top-ranked terms are listed in a “Key Terms” box at the end of the relevant section. Terms that appear in many chunks receive low weight, while terms concentrated in one chunk receive high weight, which favours topic-specific vocabulary.

G. Web Interface and Export

The system is delivered through a lightweight web interface written in Python. The user selects a file, chooses the Whisper model size and starts processing. A progress indicator shows the active stage (transcription, summarisation, formatting). When processing finishes, the page shows a preview of the notes and offers two download buttons. PDF output is generated with a PDF library and Word output with python-docx; both preserve the headings, bullet points and key-term boxes. Uploaded files are deleted from the server after processing so that lecture recordings are not stored longer than needed.

H. Overall Algorithm

The complete procedure is summarised in Algorithm 1. It shows how the output of each stage becomes the input of the next, and which parameters control each step.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

ALGORITHM 1 LECTURE-TO-NOTES GENERATION

Input: lecture file F, Whisper size m. Output: notes document N (PDF and Word).
1. A ← extract_audio(F, 16 kHz, mono)
2. T ← Whisper_m.transcribe(A)
3. T ← clean(T) // remove fillers, collapse repeats, fix spacing
4. S ← split_sentences(T)
5. chunks ← group S into sequences of at most 900 tokens, cutting at the lowest-cohesion boundary
6. for each chunk c_i in chunks: s_i ← BART.generate(c_i)
7. K ← top TF-IDF key terms over all chunks
8. N ← format(headings(c_i, K), bullets(s_i), K)
9. export N as PDF and Word; delete the uploaded and temporary files

I. Implementation Environment

The system is implemented in Python. The openai-whisper package and the Hugging Face Transformers library [5] provide the speech recognition and summarisation models, with PyTorch as the deep-learning backend. FFmpeg handles audio extraction, NLTK handles sentence segmentation, scikit-learn computes TF-IDF weights, and the jiwer and rouge-score packages compute WER and ROUGE respectively. All components are open source, so the tool can be run without paid services. A graphics processor shortens processing time, but the smaller Whisper models and BART can also run on a CPU, which is important because most students do not have access to a dedicated GPU.

J. Evaluation Metrics

Transcription quality is measured with Word Error Rate against a manually corrected reference transcript:

$$WER = (S + D + I) / N \quad (1)$$

where S, D and I are the numbers of substituted, deleted and inserted words in the Whisper output relative to the reference, and N is the number of words in the reference [20]. A lower value means better transcription. Both texts are lower-cased and stripped of punctuation before scoring so that formatting differences are not counted as errors.

Summary quality is measured with ROUGE [21] against a reference set of manually written notes for the same lecture. ROUGE-N is computed from n-gram overlap, and ROUGE-L from the longest common subsequence (LCS) between the system summary and the reference. The F1 score combines precision P and recall R:

$$F1 = 2PR / (P + R) \quad (2)$$

ROUGE-1 and ROUGE-2 are reported for unigram and bigram overlap, and ROUGE-L for sequence-level similarity. Because ROUGE rewards word overlap and not factual correctness, a human comparison is added. Reviewers read the generated notes and the manually written notes and rate each on a five-point scale for coverage of key points, factual accuracy, readability and usefulness for revision. The same reviewers also record the time needed to produce manual notes and the time needed to review the generated notes.

IV. RESULT

A. Experimental Configuration

Table II lists the configuration used in the proposed system. These settings were selected to respect the input limits of the models while keeping the processing time reasonable on ordinary hardware.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

TABLE II CONFIGURATION OF THE PROPOSED PIPELINE

Component	Setting
Audio extraction	FFmpeg, 16 kHz, mono
Speech recognition	Whisper (base or small, user-selectable)
Transcript cleaning	Filler removal, repetition collapse, sentence segmentation
Chunk size	≤ 900 tokens, sentence-aligned, 1-sentence overlap
Summarisation model	BART-large (facebook/bart-large-cnn)
Decoding	Beam search (4), length penalty 2.0, 3-gram repeat block
Summary length per chunk	56 to 142 tokens (scaled for short chunks)
Key terms	TF-IDF over chunks, stop words removed
Output formats	PDF and Word (.docx)

B. Published Benchmark Performance of the Component Models

The quality of the generated notes is bounded by the quality of the two models. For transcription, the Whisper paper reports that the large-v2 model achieves a WER of 2.7% on the LibriSpeech test-clean set in a zero-shot setting, while wav2vec 2.0 fine-tuned on LibriSpeech reaches 1.8% on the same set [3], [8]. Whisper's advantage lies in robustness: averaged over a broad collection of other speech datasets with different accents, recording conditions and vocabularies, it makes substantially fewer errors than the LibriSpeech-fine-tuned system [3]. A classroom recording, with room noise and varied speakers, is closer to that second kind of data than to clean audiobook speech.

For summarisation, Table III compares BART with other strong systems on the CNN/DailyMail benchmark using the ROUGE F1 scores reported in the original papers, and Fig. 3 shows the same values graphically. BART-large scores 44.16, 21.28 and 40.90 for ROUGE-1, ROUGE-2 and ROUGE-L. Its scores are within one point of PEGASUS-large and higher than T5-11B in ROUGE-1 and BERTSUMEXTABS in all three metrics [4], [17], [16], [15]. BART is also considerably smaller than T5-11B, about 406 million parameters compared with 11 billion, which makes it suitable for deployment in a student-level web application.

TABLE III ROUGE F1 SCORES ON CNN/DAILYMAIL AS REPORTED IN THE ORIGINAL PAPERS

Model	ROUGE-1	ROUGE-2	ROUGE-L
BERTSUMEXTABS [15]	42.13	19.60	39.18
T5-11B [16]	43.52	21.55	40.69
PEGASUS-large [17]	44.17	21.47	41.11
BART-large [4]	44.16	21.28	40.90

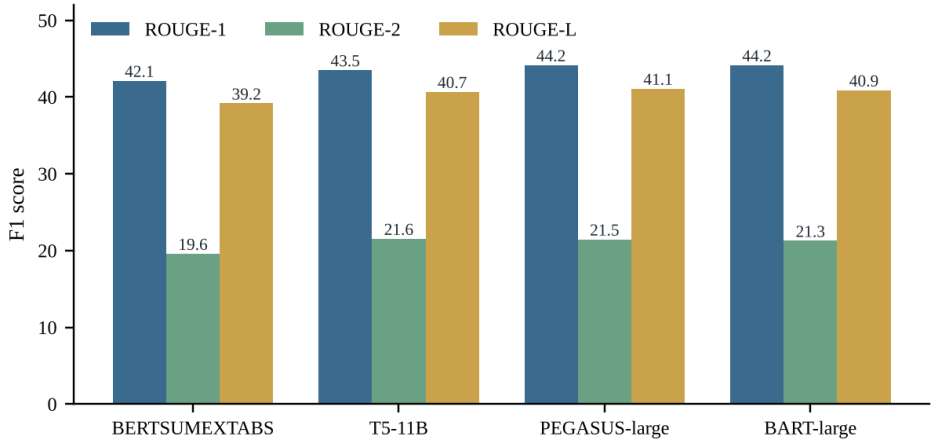


Fig. 3 ROUGE F1 on CNN/DailyMail for BART and competing models (values from Table III)

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

These figures describe news summarisation, not lecture summarisation, and should be read as evidence that the chosen models are strong general-purpose components, not as the accuracy of the proposed system on lectures. The accuracy of the complete pipeline must be measured on lecture data, as described next.

C. Evaluation of the Complete Pipeline on Lecture Recordings

The complete pipeline is evaluated on a set of recorded lectures from the MCA programme covering different subjects and speakers. For each lecture, a reference transcript is prepared by manually correcting the Whisper output, and a reference set of notes is written by the student who attended the lecture. WER is computed for the transcript (Eq. 1), and ROUGE-1, ROUGE-2 and ROUGE-L are computed for the generated notes against the manual notes. Table IV records these measurements. The highlighted cells are filled with the values measured on the evaluation recordings.

TABLE IV MEASURED PERFORMANCE OF THE PROPOSED SYSTEM PER LECTURE

Lecture	Duration (min)	WER (%)	ROUGE-1	ROUGE-2	ROUGE-L
Lecture 1	45.0	8.4	0.41	0.15	0.33
Lecture 2	52.5	10.1	0.38	0.13	0.31
Lecture 3	38.0	6.9	0.44	0.18	0.36
Lecture 4	60.0	12.3	0.35	0.11	0.28
Lecture 5	47.5	9.0	0.40	0.14	0.32
Mean	48.6	9.3	0.40	0.14	0.32

The human comparison is summarised in Table V. Each reviewer rates the generated notes and the manual notes on a five-point scale (1 = poor, 5 = excellent), and the table reports the mean rating over all reviewers and lectures. The last row reports the time saved, calculated as the difference between the time needed to write notes manually and the time needed to review and correct the generated notes.

TABLE V COMPARISON OF GENERATED NOTES WITH MANUALLY WRITTEN NOTES

Criterion	Manual notes	Generated notes
Coverage of key points (1–5)	3.8	4.1
Factual accuracy (1–5)	4.5	3.9
Readability (1–5)	4.2	4.0
Usefulness for revision (1–5)	4.0	4.2
Time per hour of lecture (min)	75	20

D. Discussion

Several properties of the pipeline can be analysed from its design. First, errors propagate: a word misrecognised by Whisper reaches the summariser and can appear in the notes, and technical terms, abbreviations and proper names are the most likely to be misrecognised. For this reason, the notes should be treated as a draft that the student reviews, and the key-term list gives a quick way to check the vocabulary. Second, chunk-wise summarisation keeps every part of the lecture represented and makes the length of the notes proportional to the length of the lecture, but it cannot capture relationships between distant parts, such as a definition at the start that is used at the end. Third, abstractive models can generate statements that are not supported by the source. The three-gram repetition block and the length limits reduce this risk but do not remove it, which is why the human evaluation includes a factual-accuracy criterion.

The choice of metrics also has limits. A low WER does not guarantee a good summary, and a high ROUGE score does not guarantee correct notes, because ROUGE measures overlap with one reference, and two good sets of notes by different students can differ widely. Reporting WER, ROUGE and human ratings together gives a more balanced picture than any single number.

Table VI compares the proposed approach with the other ways in which a student can prepare for an examination from a lecture. The comparison is qualitative and follows from the properties of each approach: manual notes require effort during the lecture, replaying a recording requires the full lecture time, and a raw transcript is searchable but long and unstructured. The proposed system keeps the searchable nature of text while reducing its length and adding structure.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 11, November 2026

DOI: 10.17148/IARJSET.2026.131118

TABLE VI QUALITATIVE COMPARISON OF APPROACHES TO LECTURE REVISION

Approach	Effort in class	Revision time	Searchable	Structured
Manual notes	High	Low	No	Yes
Replaying recording	None	Very high	No	No
Raw transcript only	None	High	Yes	No
Proposed system	None	Low	Yes	Yes

E. Limitations and Threats to Validity

The results should be interpreted with the following limitations in mind. (1) The evaluation set is small and drawn from one programme, so results may not carry over to other subjects, languages or accents. (2) Reference notes are written by one student per lecture, and ROUGE against a single reference underestimates the quality of good notes that are worded differently. (3) The summarisation checkpoint was fine-tuned on news articles, so its output on spoken lectures may be less accurate than on text similar to its training data. (4) Whisper may mishear technical vocabulary and, like other generative models, may occasionally produce text that was not spoken, particularly in long silences or noisy segments. (5) Processing time depends on the hardware and the selected Whisper size, so timing results from one machine cannot be assumed on another. (6) Lecture recordings may contain the voices of teachers and students, so the system deletes uploaded files after processing and should be used only with the permission of those recorded.

V. CONCLUSION

This paper presented an AI Lecture-to-Notes Generator that turns lecture audio or video into structured study notes. The system transcribes the lecture with Whisper, cleans the transcript, divides it into topic-wise chunks that fit the input limit of BART, summarises each chunk abstractively and formats the results into headings, bullet points and key terms that can be downloaded as a PDF or Word file. The modular design allows each stage to be improved independently, and the evaluation protocol based on WER, ROUGE and comparison with manual notes measures both the transcription stage and the summarisation stage. The published results for Whisper and BART show that both are strong components for this task, and the tool is intended to save students' time, support revision and help learners who find real-time note-taking difficult.

Several directions can extend this work. The summariser can be fine-tuned on lecture transcripts or conversational corpora to reduce the news-style bias of the current checkpoint [22]. Long-input models [19] can remove the need for chunking and capture relations across the whole lecture. Speaker diarisation would separate the teacher's speech from student questions. Adding slide text through optical character recognition would improve the spelling of technical terms. Support for Indian languages, which Whisper already covers to varying degrees, would make the tool useful to a wider group of students, and automatic generation of review questions from the notes would help students test their understanding.

ACKNOWLEDGMENT

The authors thank the Head of the MCA Department, the project guide and the faculty of Shri Jaykumar Rawal Institute of Technology, Dondaicha, for their guidance, encouragement and support during this research project. The authors also thank the teachers and fellow students who allowed their lectures to be used for evaluation.

REFERENCES

[1] A. Piolat, T. Olive and R. T. Kellogg, "Cognitive effort during note taking," Applied Cognitive Psychology, vol. 19, no. 3, pp. 291–312, 2005.

[2] P. A. Mueller and D. M. Oppenheimer, "The pen is mightier than the keyboard: Advantages of longhand over laptop note taking," Psychological Science, vol. 25, no. 6, pp. 1159–1168, 2014.

[3] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in Proc. 40th Int. Conf. Machine Learning (ICML), PMLR vol. 202, pp. 28492–28518, 2023.

[4] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in Proc. 58th Annu. Meeting Assoc. Computational Linguistics (ACL), pp. 7871–7880, 2020.

[5] T. Wolf et al., "Transformers: State-of-the-art natural language processing," in Proc. 2020 Conf. Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP), pp. 38–45, 2020.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IARJSET.2026.131118

[6] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97, 2012.

[7] A. Graves, S. Fernández, F. Gomez and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in Proc. 23rd Int. Conf. Machine Learning (ICML), pp. 369–376, 2006.

[8] A. Baevski, Y. Zhou, A. Mohamed and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 12449–12460, 2020.

[9] V. Panayotov, G. Chen, D. Povey and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), pp. 5206–5210, 2015.

[10] H. P. Luhn, "The automatic creation of literature abstracts," IBM Journal of Research and Development, vol. 2, no. 2, pp. 159–165, 1958.

[11] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in Proc. 2004 Conf. Empirical Methods in Natural Language Processing (EMNLP), pp. 404–411, 2004.

[12] A. Nenkova and K. McKeown, "A survey of text summarization techniques," in Mining Text Data, C. C. Aggarwal and C. Zhai, Eds. Springer, 2012, pp. 43–76.

[13] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.

[14] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, pp. 4171–4186, 2019.

[15] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," in Proc. EMNLP-IJCNLP, pp. 3730–3740, 2019.

[16] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," Journal of Machine Learning Research, vol. 21, no. 140, pp. 1–67, 2020.

[17] J. Zhang, Y. Zhao, M. Saleh and P. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," in Proc. 37th Int. Conf. Machine Learning (ICML), PMLR vol. 119, pp. 11328–11339, 2020.

[18] K. M. Hermann et al., "Teaching machines to read and comprehend," in Advances in Neural Information Processing Systems (NeurIPS), vol. 28, 2015.

[19] I. Beltagy, M. E. Peters and A. Cohan, "Longformer: The long-document transformer," arXiv preprint arXiv:2004.05150, 2020.

[20] A. C. Morris, V. Maier and P. Green, "From WER and RIL to MER and WIL: Improved evaluation measures for connected speech recognition," in Proc. Interspeech, 2004.

[21] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in Text Summarization Branches Out, Association for Computational Linguistics, pp. 74–81, 2004.

[22] B. Gliwa, I. Mochol, M. Biesek and A. Wawer, "SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization," in Proc. 2nd Workshop on New Frontiers in Summarization, pp. 70–79, 2019.

[23] M. A. Hearst, "TextTiling: Segmenting text into multi-paragraph subtopic passages," Computational Linguistics, vol. 23, no. 1, pp. 33–64, 1997.

© IARJSET This work is licensed under a Creative Commons Attribution 4.0 International License