Context-Aware Speech Transcription and Q&A System

Incorporating ML Models in a New Use Case Setting

YASH MATHUR, University at Buffalo, USA
VARUN SINGH, University at Buffalo, USA
VISHWAS REDDY MALI, University at Buffalo, USA

ACM Reference Format:
Yash Mathur, Varun Singh, and Vishwas Reddy Mali. 2025. Context-Aware Speech Transcription and Q&A System: Incorporating ML Models in a New Use Case Setting. 1, 1 (April 2025), 9 pages.

1 ABSTRACT

This project proposes a Context-Aware Speech Transcription and Question Answering (Q&A) System that combines Large Language Model (LLM) and Automatic Speech Recognition (ASR) technologies to facilitate effective, end-to-end transcription and chat. Llama 3.2 1B from Meta and Whisper Tiny from OpenAI which are lightweight models were finetuned on the SQuAD and LibriSpeech datasets, respectively, to improve transcription accuracy and contextual understanding under resource-constrained environments. A modular approach was followed where an audio file (upto 100 MBs) is fed into the ASR model which outputs the transcription which is then taken as contextual input in the LLM. This entire system is hosted locally through a Streamlit/Flask based application allowing users to upload audio, query, and receive answers with session data being saved using SQLite along with a login system. The most important results indicated that the Whisper Tiny finetuned model achieved a lower word error rate (WER) of 18.41% over a baseline of 19.10%, showing a 4% increase in performance, while the finetuned Llama 3.2 1B model generated more fluent answers with improved BLEU and perplexity scores, where BLEU scores were increased from 0.134 to 0.248, however, showed a slight decline in F1 and adaptability to shuffled contexts. The tested application accurately transcribed and asked questions from a 15-minute audio input in minutes, confirming the responsiveness and usability of the system. Such results highlight the feasibility of releasing small, highly optimized models within an integrated framework to offer robust speech and language understanding in compact environments.

2 INTRODUCTION

The advent of voice technology has revolutionized how users engage with computational systems. From smart home applications and health care dictation devices, learning environments and education transcript tools, to customer service bots, voice-based interfaces have become an inseparable feature of user interactions today. Despite colossal progress across automatic speech recognition (ASR) and large language models (LLMs), there still exists a set of issues once they are deployed across real-world and unscripted contexts. Accents, background noise, disfluencies, and shifting conversation contexts often lead to degraded performance in both transcription accuracy and subsequent language understanding tasks.

To overcome such limitations, this study aims to develop a Context-Aware Speech Transcription and Q&A System that augments ASR and LLMs to enable end-to-end voice communication. Allowing users to input speech or audio, get correct transcriptions enabling context support, and engage in a subsequent Q&A session with the system is intended to simulate a natural conversation experience.

2.1 Landscape and Relevance

The recent progress of transformer models and end-to-end learning approaches has caused an abrupt shift of the entire NLP and ASR research community. OpenAI's Whisper open-source ASR model has seen dramatic gains in transcription accuracy in a broad range of languages and applications. The synchronization of the ASR system with LLM systems is yet not simple to achieve, however, despite better developments in such lines.

Also, availability of extremely large amounts of high-quality data and computational resources is assumed for the majority of state-of-the-art systems. Though effective, alternatives like SpeechBERT are computationally intensive and unsuitable for situations where lightweight deployment is paramount. Hence, resource-aware, scalable system development capable of executing reliably across a broad spectrum of application domains without accessing costly APIs or proprietary systems is still missing.

To bridge this shortfall, our project exploits domain-specific data to improve upon open-source models such as Llama 3.2-1B and Whisper Tiny, which are light but effective. Through the use of meticulous training strategy planning and pipeline integration, we aspire to demonstrate that the aforementioned systems can be competitive even under situations of low computational resources, thereby making them more practical for small companies, schools, and university researchers.

2.2 Motivation and Use Case Context

The project motivation lies in the immediate need for context-sensitive speech interfaces in applications where responsiveness, intelligence, and accuracy are crucial as well as resource constraints require the use of lighter models. In such applications, an accurate system that transcribes verbal records and allows the user to search

Authors' addresses: Yash Mathur, University at Buffalo, Buffalo, NY, USA, ymathur@buffalo.edu; Varun Singh, University at Buffalo, Buffalo, NY, USA, vsingh35@buffalo.edu; Vishwas Reddy Mali, University at Buffalo, Buffalo, NY, USA, vmali@buffalo.edu.

ACM XXXX-XXXX/2025/4-ART
https://doi.org/

relevant segments of the transcript can greatly improve documentation workflows.

In studying, students increasingly rely on recorded lectures and machine-generated transcripts for studying course material. Augmenting such transcripts with a Q&A feature allows students to interact actively and at their own pace with material, to become more intensely engaged and on an individual basis. Intelligent personal assistants, normally executed in noisy environments or by speakers with different accents, also benefit greatly from systems that can deal with disfluent, context-dependent utterances and provide correct and coherent answers.

These application scenarios demand the creation of systems that go beyond surface transcription and instead engage in semantic understanding. By creating a modular and extensible pipeline to facilitate such abilities, our project aims to empower the user in a variety of real-world applications.

2.3 Overall Plan and Expected Contributions

The overall plan for this project is composed of two distinct yet interconnected phases. Phase 1 involves fine-tuning the LLM and ASR models on task-appropriate datasets. Whisper Tiny is fine-tuned on the LibriSpeechcorpus to enhance its generalization ability across speakers, accents, and background conditions. Meanwhile, Llama 3.2-1B is fine-tuned on SQuAD v1.1, a reading comprehension and QA benchmark. These steps are laboriously optimized with parameter-efficient techniques such as LoRA, utilizing available resources on Google Colab Pro with CUDA acceleration to manage computational overhead.

Phase II is committed to pipeline integration, in which the output of the ASR model is passed as context to the LLM for question-answering. Prompt engineering is used on the pipeline for making the model responses both contextually accurate and conversationally natural. Such modular architecture lends flexibility in deployment, allowing future upgrades or dataset extensions.

The anticipated contributions of this project are two-fold. On the technical front, we demonstrate that dramatic improvements in transcription and Q&A performance can be achieved through diligent fine-tuning of compact, open-source models. On the architectural front, we provide a design pattern by building a minimal viable application working on the voice-interaction systems that can be operated in environments with modest computational resources.

2.4 Significance in the Current Technological Landscape

This project lies at the intersection of speech recognition, contextual NLP, and low-resource machine learning deployment. It addresses some of the open issues in ASR and LLM integration, including context preservation, error robustness, and response relevance. In this manner, it directly confronts the limitations of existing systems that are either too computationally demanding or too weak for domain-specific applications.

With its deliberate design choices—prioritizing resource efficiency, open-source leverage, and adherence to realistic constraints—this project contributes practically and usefully to voice-based AI research.

3 LITERATURE REVIEW

Automatic Speech Recognition (ASR) and Spoken Question Answering (SQA) systems have undergone a dramatic transformation in recent years. Earlier ASR systems were dominated by statistical approaches such as Hidden Markov Models (HMMs) combined with Gaussian Mixture Models (GMMs), which relied heavily on handcrafted features and domain-specific engineering. These traditional systems often required carefully tuned phonetic dictionaries and complex alignment algorithms. The advent of deep learning models, particularly Deep Neural Networks (DNNs) and Recurrent Neural Networks (RNNs), revolutionized the field by enabling end-to-end training from raw audio to text [1, 2].

Despite these advancements, traditional ASR systems generally followed a pipeline structure, where audio is transcribed into text using an acoustic model, and the transcribed text is then processed by a language model for downstream tasks like question answering or intent classification [4, 12]. This modular approach, while interpretable, introduced cascading errors—where mistakes in early components (e.g., transcription) propagate through the pipeline, reducing overall system accuracy.

To overcome these limitations, researchers began exploring end-to-end ASR systems, which jointly optimize the acoustic and language modeling stages in a single architecture. Examples include encoder-decoder models, Connectionist Temporal Classification (CTC), and RNN-Transducers [1, 3, 14]. These architectures eliminate hand-engineered feature extraction and allow models to learn directly from raw waveform inputs, thereby improving robustness and generalization. However, challenges exist, particularly with rare-word recognition, domain adaptation, and resource constraints.

A promising strategy for improving ASR accuracy is through context-aware modeling. These models integrate external knowledge such as user history, named entities, topic embeddings, or visual context to refine both acoustic and language predictions [1, 8]. For example, Lakomkin et al. introduced Speech-Llama, which reframes ASR as a mixed-modal decoding task by prepending textual context tokens to the audio input, reducing Word Error Rate (WER) significantly on rare-word benchmarks [8]. Similarly, Sudo et al. proposed bias phrase beam search, injecting pre-defined domain-specific terms into beam decoding using an attention-based mechanism, which improves ASR accuracy without retraining [12].

Nonetheless, many of these context-aware systems are computationally expensive and rely on structured inputs or curated phrase sets, making them impractical for scalability in low-resource environments [10, 11]. In contrast, our proposed approach is centered around parameter-efficient fine-tuning of the Whisper Medium ASR model, trained on the LibriSpeech dataset. Whisper, developed by OpenAI, provides multilingual and multitask capabilities, and our fine-tuning specifically improves performance on noisy audio and low-frequency vocabulary—two core weaknesses of conventional ASR systems.

Simultaneously, progress in Spoken Question Answering (SQA) has paralleled ASR innovation. Traditional SQA systems relied on high-quality ASR outputs before applying a natural language question-answering model. However, this dependency introduces vulnerabilities to transcription errors. Models like SpeechBERT and

ST-BERT address this by learning joint audio-text representations that reduce error sensitivity. These models operate directly on acoustic embeddings, allowing them to bypass some transcription limitations and handle informal, accented, or noisy speech better [6, 17].

Our contribution to SQA involves fine-tuning Llama 3.1 [8] on the SQuAD dataset, integrating it into the pipeline following Whisper-based transcription. The system works by transcribing the spoken question via Whisper, then tokenizing the output and feeding it into the Llama model, which retrieves or generates the best-matching answer from context. By optimizing both components independently and ensuring seamless integration at inference, the system balances accuracy with speed. The architecture is intentionally lightweight, designed to run within resource-constrained environments like Google Colab.

The integration of multimodal inputs—such as visual features from video or images—has also advanced both ASR and SQA tasks. For instance, Gupta et al. demonstrated that acoustic and language models could be adapted using visual embeddings (e.g., objects like "car", scenes like "kitchen") to improve transcription accuracy in YouTube-style content [3]. Similarly, Hori et al. introduced an audio-visual scene-aware dialog system, which uses video frames and dialog history to generate more accurate and relevant responses [5]. While these approaches enhance performance in rich media settings, they typically require large-scale datasets and high-end hardware.

To address these computational challenges, we focus on unimodal but context-aware architectures that do not rely on simultaneous video streams. However, we do draw methodological influence from foundational multimodal models like CLIP by Radford et al., which learn aligned embeddings across modalities [11].

Another key area of development lies in rescorer and post-processing models. Wei et al. proposed RNNLM lattice rescoring, incorporating contextual signals such as speaker turn and dialog topic to improve hypothesis ranking [14]. Ortega et al. combined CNNs with CRFs to classify dialog acts from ASR outputs and showed that even noisy transcripts from end-to-end ASR models could outperform hybrid system outputs on downstream dialog tasks [10].

In the realm of SQA evaluation, Zhao et al. introduced LibriSQA, a benchmark dataset designed to test spoken QA systems using large language models. It enables direct comparison across models that process spoken input versus text-only baselines [17]. For acoustic-level fusion, Kuo et al. presented a BERT-based multiple-choice QA system that integrates raw speech embeddings alongside text to mitigate the effect of ASR errors [7].

Finally, Xiong et al. achieved a milestone in the field by demonstrating human parity in ASR using ensemble modeling and lattice-free MMI training [15]. Their work provides a high-performance ceiling and validates the feasibility of building ASR systems that match or exceed human transcriptionists under specific conditions. Complementing this, Liao et al. focused on enhancing ASR readability, treating it as a sequence-to-sequence problem to produce grammatically correct and human-friendly outputs [9].

These advancements reveal a shared goal across ASR and SQA research: improving robustness, minimizing error propagation, and reducing computational overhead. Our system, which combines a fine-tuned Whisper Medium model with a Llama 3.1-based SQA module, is designed to address these priorities. By operating efficiently in noisy, domain-shifted, and resource-constrained settings, it advances the usability and scalability of real-time spoken language understanding systems.

4 DATASETS

The development of a chatbot with integrated large language model (LLM) and automatic speech recognition (ASR) capabilities was dependent on the selection of robust, diverse, and complex data. This project uses the Stanford Question Answering Dataset (SQuAD 2.0) to train and test the LLM capability for contextual understanding and question-answering, and LibriSpeech to develop the ASR capability for accurate speech-to-text transcription. Both datasets are widely recognized in their fields, having rich, real-world problems which the project goal of creating a multi-modal dialogue system can suit. This section provides a detailed examination of the dataset structures, why they are useful to the chatbot, their past uses, their availability on all platforms, and special challenges encountered when integrating them.

4.1 Overview of Selected Datasets

SQuAD 2.0 is an excellent natural language processing (NLP) data source that assists in quantifying a model's ability to understand text as well as extract answers. The dataset includes more than 150,000 question-answer pairs from Wikipedia articles and includes both answerable and unanswerable questions and, hence, is ideal to train the chatbot's LLM on diverse user requests, including ambiguous ones with no suitable context. LibriSpeech, a 1,000-hour corpus of audiobook recordings of English speech, provides high-quality, diverse speech data to train the ASR system, enabling robust transcription of user inputs under clean and noisy conditions. Such datasets were chosen for their recentness, richness, and conformance to the project's two objectives of text-based reasoning and speech processing.

In addition, SQuAD and LibriSpeech datasets were used so as to respond to the requirement for datasets having realistic conversational complexities like indistinct or unanswerable questions and speech of diverse accents or background noise. Both datasets are available on various platforms, i.e., dedicated websites, Hugging Face, Kaggle, and machine learning libraries, so as to integrate them perfectly into modern pipelines. However, their large sizes, complex structure, and inherent variability present significant preprocessing and computational challenges that need to be addressed by specialized approaches. The subsequent sections take into account the specifics of each dataset and the overall challenge they present.

4.2 SQuAD for LLM Finetuning

SQuAD 2.0 is a reading comprehension dataset with 130,319 training questions and 11,873 development questions, taken from over 500 Wikipedia articles. Each example is composed of a context paragraph (typically 100–500 words), a question, and an answer, as a character-level start and end indices text span or a null answer for some 50,000 unanswerable questions. The data is provided in JSON format, hierarchically organized: an outer list of articles contains titles and paragraphs, and each paragraph includes a context and an

inner list of question-answer pairs. This framework allows for fine-grained testing of contextual comprehension, answer extraction, and the determination of unanswerable questions. The variety of topics, from history to science, guarantees broad coverage, while the addition of unanswerable questions introduces complexity essential to practical applications.

The data follows a sample structure as follows:

```
SQuAD_dataset/
|-- train-v1.1.json
|   |-- data
|   |   |-- title
|   |   |-- paragraphs
|   |   |   |-- context
|   |   |   |-- qas
|   |   |   |   |-- id
|   |   |   |   |-- question
|   |   |   |   |-- answers
|   |   |   |   |   |-- text
|   |   |   |   |   |-- answer_start
```

SQuAD 2.0 is extremely relevant to the LLM capability of the chatbot because it tests the model explicitly in its ability to handle long text and responding with appropriate, contextually relevant answers. Incorporating unanswered questions mimics actual usage when users pose questions with insufficient context, and the chatbot must offer reasonable replies (e.g., "I don't know enough to answer that"). The broad variety of topics and magnitude in the data are in accordance with the intended use of the chatbot as a general-purpose conversation aid, as well as facilitating easier fine-tuning of transformer models. Through training on SQuAD, our LLM (Llama 3.2-1B) can achieve strong performance on question-answering tasks, which is among the primary functions of the text-based conversation part of the chatbot.

We also looked at past usage contexts and citation of the SQuAD dataset, published in 2018, SQuAD 2.0 has served as an NLP anchor dataset, serving as a baseline for today's best-of-the-breed models. It played a crucial role in the training of BERT, which achieved human-level performance on exact match (EM) and F1 score, and subsequent models like RoBERTa and T5. SQuAD has been utilized to advance transfer learning in domain-specific applications, such as medical question-answering (e.g., BioASQ) and legal document analysis, demonstrating its applicability. Its unanswerable questions have also spurred research for uncertainty estimation and abstention methods, critical for reliable conversational systems. The fact that the dataset is being utilized across different applications demonstrates that it is stable enough to quantify LLM performance.

Availability and Integration: The SQuAD dataset is officially available at webpage (https://rajpurkar.github.io/SQuAD-explorer/) provides training and development sets in JSON files, while the Hugging Face datasets library (`load_dataset("squad_v2")`) provides programmatic access. The data is also available on Kaggle and has been included in the `transformers` library for fine-tuning pre-trained models. The websites provide ease of integration within Python-based NLP pipelines, thus making SQuAD a vital component of academic projects. The compact size of the dataset (around 50 MB) facilitates its storage and processing, although its complexity requires special treatment.

4.3 LibriSpeech for ASR model Finetuning

LibriSpeech is a large corpus of approximately 1,000 hours of English audiobooks sampled at 16 kHz and divided into "clean" (460 hours, high signal-to-noise) and "other" (500 hours, noisy) subsets and others. It has seven subsets: train-clean-100, train-clean-360, train-other-500, dev-clean, dev-other, test-clean, and test-other, containing approximately 2.5 million utterances. Audio files are in .flac format, transcriptions as .txt format, and hierarchically stored by speaker and chapter IDs. Metadata files hold speaker information (e.g., gender, accent, recording environment) and book metadata. Transcriptions are in uppercase letters and without punctuation, emulating spoken language behavior. The variety of speakers (over 2,000) and acoustic environments present in the dataset make it an useful training material for ASR.

LibriSpeech is better suited for our chatbot's ASR ability because its versatile speech data allows user inputs to be converted from a wide number of accents, speaking modes, and environmental settings. This also has the added advantage during the process of OOD Robustness testing of ASR systems. The noise and clean subsets allow for the creation of a strong model that can function in actual conditions, such as on mobile phones or with background noise. With LibriSpeech, the chatbot can achieve good transcription accuracy, a necessary precursor for smooth voice interaction.

Our research also found LibriSpeech dataset to have taken a central place in ASR research ever since it was made public in 2015. LibriSpeech has been used as a source of standard and comparison in ASR research and has promoted models like Deep Speech, Wav2Vec, and Whisper. The models achieved significant word error rate (WER) improvement and have recorded unprecedented WER less than 5% in the clean subsets by Whisper. LibriSpeech has supported the research into noise-robust ASR, end-to-end speech recognition, and low-resource language transfer learning. Applications of LibriSpeech are voice assistants (Siri, Alexa), transcription applications, and systems for hearing-aid users' accessibility. Its clean set and noisy set have both been utilized in acoustic variability studies and have therefore been instrumental in creating a groundwork for ASR robustness systems.

Availability and Integration: LibriSpeech is available to download for free at OpenSLR (https://www.openslr.org/12/), where one may also download the full 60 GB corpus. It can also be used via Hugging Face's library (`load_dataset("librispeech_asr")`), torchaudio, and the `transformers` library, so integration into PyTorch-based pipelines is straightforward. Partial sets are also present on Kaggle, but the full set usually originates from OpenSLR due to its magnitude. The open dataset license and the backing of platforms ensure mutual access for development and research, though it is too large to be realistically workable.

4.4 Data Challenges and Task Complexity

Both SQuAD and LibriSpeech contain high complexity that is seriously challenging and emphasizes their application in the chatbot project.

For SQuAD, its hierarchical JSON structure required us to author custom preprocessing scripts to create context-question-answer triplets, particularly for unanswerable questions, which had to be handled specially within the training pipeline. Memory constraints during preprocessing rendered batch processing inevitable, with caching the whole dataset (150,000+) in memory slowing down our humble computational hardware. The range of question types—factual, inferential, and unanswerable—called for hyperparameter tuning and tailored loss functions to maximize answer prediction and accuracy in null responses. For example, we employed a weighted cross-entropy loss to prefer correct null predictions, addressing the dataset-specific issue of unanswerable questions. In the case of LibriSpeech, the 1,000-hour corpus likewise created major storage and computation problems, which led to distributed preprocessing on a Google Colab high-performance computing machine (A100).

Retrieving the dataset using Hugging Face's API both locally and on Colab posed a challenge due to rate-limited, with fallbacks to direct downloads from OpenSLR, making the pipeline configuration complicated. The hierarchical directory structure introduced alignment issues, with some audio files lacking matching transcriptions due to corruption or metadata issues, necessitating robust error-checking scripts. The loud "other" subsets of speakers reduced WER, particularly for strongly accented speakers or low-quality media, and required advanced data augmentation techniques (e.g., noise injection, speed perturbation) to improve the model's performance, which presented a major challenge and was removed eventually to stay in scope of the project. Post-processing had to be added because transcriptions did not include punctuation to allow sentence boundaries in order to make it possible to match the input format of text by the LLM. Such challenge tasks reflect actual complexity of datasets such as working with unclear questions (SQuAD) and transcribing dirty, mixed-up speech (LibriSpeech). They required complex preprocessing pipelines, management of resources, and tuning in domain such as parallelization for LibriSpeech and tokenization for SQuAD.

This research can further be accelerated by using automated preprocessing pipelines and domain-specific fine-tuning to further mitigate these issues.

5 APPROACH

The development of a text-based conversational chatbot that utilizes automatic speech recognition (ASR) to transcribe audio inputs and a large language model (LLM) to provide contextually relevant responses requires a well planned and methodological approach that leverages the Stanford Question Answering Dataset (SQuAD 2.0) and LibriSpeech datasets. This project employs the Llama 3.2 1B model for LLM and Whisper Tiny model for ASR, which will be fine-tuned on small subsets to deliver optimal performance under computational limitations. Preprocessing, baseline testing, model configuration, fine-tuning, comparison analysis, model selection, output extraction, and deployment through a locally run Streamlit application are all stages of the project. Additionally, heteroscedasticity, double descent, and out-of-distribution robustness tests along with other key metric will be used to measure model performance over a variety of inputs (Following Milestone 1 feedback that close examination was needed). This section describes each step, problems and scalability suggestions for improved performance with more adequate resources.

5.1 Overview of the Approach

The development of the chatbot began with strategic model choice for performance vs. computation trade-off. For ASR, we employed the Whisper Tiny (openai/whisper-tiny, ~39M parameters) model due to its sparse architecture designed for low-resource environments like Google Colab's and capability to train on limited GPU, and robust pre-training on multi-speaker audio data, thereby making it ideal for transcribing LibriSpeech's clean audio. Llama 3.2 1B (meta-llama/Llama-3.2-1B, ~1B parameters) was chosen for the LLM's question-answering performance because of its transformer model and ability to enable LoRA (Low-Rank Adaptation) for SQuAD 2.0 fine-tuning. These models were chosen over their larger counterparts (e.g., Llama 3.2 11B, Whisper Large) because they suited our memory restrictions during training and inference and offered a useful starting point for fine-tuning.

The Whisper Tiny Model extracts outputs by converting audio inputs into mel-spectrograms, processed through the model to generate transcriptions. This is implemented in the ASR code provided. Further, the WhisperProcessor tokenizes transcriptions, and the model's generate method produced text outputs, decoded to lowercase strings for consistency with LibriSpeech's format. For Llama 3.2 1B, outputs are extracted by feeding tokenized context-question prompts (implemented in the LLM code) into the model, using greedy decoding to generate short, precise answers. The answer text can then be extracted once this processing is over, which filters out non-ASCII characters and irrelevant lines (e.g., "Context:"), ensuring compatibility with SQuAD's answer format.

To manage computational limits, 5,000-instance SQuAD 2.0 training set was selected, and a 3-hour subset (2155 paired instances) was extracted from a 100-hour LibriSpeech train-clean-100 subset. Preprocessing involved extensive data transformations, including tokenization and format standardization for SQuAD, and transcription normalization, and file pairing for LibriSpeech. Then, the Models were configured with Hugging Face's transformers and unsloth libraries, adapting LoRA for efficient fine-tuning, as preparation for the finetuning stage. The approach also included measuring the baseline evaluations (WER, CER, SeMaScore for ASR; F1, EM, BLEU, ROUGE, METEOR for LLM) at this stage. Later, fine-tuning was done, with derived loss and performance metrics to track the finetuning effectiveness. Following this, we performed comparative analysis, and local deployment via a Streamlit frontend and a flask based backend. These steps are described in detail in the further sections.

Evaluation metrics were also chosen carefully, for their alignment with LibriSpeech (WER, CER, SeMaScore for transcription accuracy and semantic fidelity) and SQuAD (F1, EM for answer precision, plus BLEU, ROUGE, METEOR for text similarity), expanded per Milestone 1 feedback to include text similarity metrics and OOD robustness for varied inputs. Heteroscedasticity (error variance among speakers/contexts) and double descent (performance patterns with

training stages) were also investigated to enhance robustness. Colab's A100 GPU was needed for fine-tuning because of the simultaneous coding and training, which used over 200 compute units during the project. To make scaling easier, dependency conflicts and dataset issues were resolved discussed in further sections.

5.2 Preparing and preprocessing data

Preprocessing evolved from Milestone 1's inefficient local downloads to speed up data access. We used Hugging Face's datasets library (load_dataset("rajpurkar/squad_v2", split="train[:5000]")) to import a 5,000-instance training subset directly in order to fine-tune SQuAD 2.0 to match question types (factual, inferential, and unanswerable) with the LLM code. The subsequence was tokenized using Llama 3.2 tokenizer, preformatting prompts as "Context: ... Question: ... Answer: ...", and split into 80% train (4000 examples) and 20% validation (100 examples). A 5-example validation set was treated similarly, with OOD data created by shuffling context sentences. This approach bypassed Milestone 1's JSON parsing issues, though unanswerable questions required special null response encoding, as described under the dataset section.

For LibriSpeech, we fetched a 100-hour train-clean-100 tar file from OpenSLR and unpacked it to Google Drive, as contrasted with the local processing in Milestone 1. A 3-hour subset (around 58 utterances and 5 speakers) was retrieved by the ASR code and stored in /content/librispeech_subset. This is equivalent to 2,155 training files and 261 test files. torchaudio was used to resample audio files (.flac) at 16 kHz, and postprocessing was used to add sentence boundaries to transcriptions (.txt). A validation script was employed to resolve extraction issues (around one mismatched flac). This small subset addressed Colab's memory constraints and the base Whisper Tiny's poor generalization.

5.3 Baseline Assessments

Baseline tests were carried out on the pre-trained, unaltered Whisper Tiny and Llama 3.2 1B models in order to establish performance benchmarks post finetuning. The feedback from Milestone 1 was incorporated for a comprehensive analysis. The code implemented (submitted) Whisper Tiny for ASR using measures such as word error rate (WER), character error rate (CER), and SeMaScore on a 0.6-hour test set of the LibriSpeech dev-clean-test subset. These metrics were chosen because they are standard in speech recognition research: WER and CER challenge how accurately words and characters are transcribed, and SeMaScore challenges how accurately the meaning of the transcription matches the original text by comparing sentence embeddings through a pre-trained model. This is in line with Milestone 1 feedback on including more robust analysis. An OOD test was also run by adding small noise (0.01 * randn) to the test audio through the add_noise function, testing the model's behavior against real-world audio variations. To investigate error deviation between speakers with different speaking styles, per-speaker metrics were also investigated which further improved OOD analysis.

For LLM a 5-instance validation set from SQuAD 2.0 was used to derive evaluation metrics on the baseline (pretrained) Llama 3.2-1B model. These evaluation metrics included F1 score, exact match (EM), BLEU, ROUGE-1, ROUGE-L, and METEOR. These were selected to match SQuAD's evaluation standards: F1 and EM, gauge how well answers match the target responses, and BLEU, ROUGE, and METEOR gauge how well the generated text matches the correct answer to provide a more holistic view. Additonally, an OOD test was also conducted by using shuffled context sentences through the preprocess_ood function to test the model's capacity to handle disturbed context. Error variances across different context lengths were examined for understanding performance stability, and performance trends were monitored for determining whether there were initial dips followed by recoveries. All tests were run on Google Colab's A100 GPU for consistency with other fine-tuning processes.

5.4 Model Preparation for Fine-Tuning

Model preparation and fine tuning used code from an article on Hugging Face discussing steps to fine tune Llama 3.2-1B model as the base, as well as the code from Hugging face on unsloth for LLM fine tuning [13, 16]. For Llama 3.2 1B, we had to get access via Hugging Face as the model is gated, and later required a token, loaded with unsloth (FastLanguageModel.from_pretrained) to be used. The LLM code adapted a LoRA script, using 4-bit quantization and a 256-token sequence length. Dependency conflicts with bitsandbytes were resolved by allowing pip's build wheel to handle conflicts, The tokenized SQuAD subset was then prepared for LoRA training.

For Whisper Tiny, the model was loaded via transformers (WhisperForConditionalGeneration.from_pretrained), using an ASR fine-tuning script. The 3-hour LibriSpeech subset was processed into input features. Noise injection augmented data, addressing the base model's limitations. Both models were prepared on Colab's A100 GPU.

5.5 Fine-Tuning Process

Fine-tuning used the provided code. For Llama 3.2 1B, the 5,000-instance training subset was fine-tuned using LoRA (r=16, alpha=32, target_modules=["q_proj", "v_proj"]), with 3 epochs, learning rate 2e-5, batch size 4, gradient accumulation steps 4, warmup steps 50, weight decay 0.01, and fp16.

Using LoRA (r=8, alpha=16), 400 steps, learning rate 1e-5, batch size 2, warmup steps 50, and fp16, the 3-hour training subset was optimized for Whisper Tiny.

5.6 Integration and Application Development

The pipeline integrates ASR and LLM, with Whisper Tiny for audio transcription and Llama 3.2 for response generation. A Streamlit app provides a text-based interface, supporting audio upload (pagination if file greater than 30 seconds), text question input, and response display. Features include login (A simple login with credentials stored locally on a database), session management (allowing for multiple transcription sessions without the need for logging out), and SQLite storage of the trascriptions. The frontend uses streamlit, and the backend, a Flask script, handles transcription and answer generation. The app was tested on a Windows 11 machine (32 GB RAM, Intel Core 9 Ultra).

5.7 Scalability and Performance Enhancement

The current setup, limited by Colab's resources and cost, as well as unavailability of GPU locally, can be extended with advanced infrastructure to achieve superior post-training performance. The project can be extended to use AWS SageMaker with NVIDIA A100 or V100 clusters, which would enable training on the full SQuAD (130,319 instances) and LibriSpeech (1,000 hours) datasets, improving generalization and reducing heteroscedasticity much more. Distributed frameworks like Horovod could support larger batch sizes (16–32), accelerating convergence. Moreover, improved baseline models such as Whisper Medium or Whisper Large for ASR would enhance transcriptional accuracy due to their power of pre-training, especially in noisy speech. Llama 4, or Llama 3.2 11B for LLM, will heavily enhance question-answering by employing deeper models to serve heavy SQuAD workloads while maintaining efficiency with LoRA. These models, executed on the computing strengths of AWS, would stabilize training, minimizing double descent. In addition, hosting the Streamlit app on AWS Elastic Beanstalk or ECS would facilitate high-traffic scenarios, where AWS Lambda would have transcription and answer generation as serverless operations for low latency. Drive's 15 GB limit would be circumvented by storing datasets and checkpoints in S3 buckets. Route 53/CloudFront would provide global low-latency access, while ElastiCache with Redis would employ session caching. To further enhance WER and F1, SageMaker's Hyperparameter Optimization would also adjust LoRA ranks and learning rates. As a result, the chatbot would be very effective, scalable, and deployable for practical uses.

5.8 Challenges

The project encountered a number of challenges, including robustness tests, Colab quotas, data access, insufficient preparation, scoring computation, and dependency conflicts. In the use of Llama 3.2 1B, dependency conflicts that largely occurred due to bitsandbytes, which is required for 4-bit quantization for simplicity in training. These were fixed by using torch==2.1.0 and transformers==4.36.2 to resolve the conflict. Access to the gated Llama model was also initially an issue, with one of the team members being rejected for the access initially, which was later received. Also, downloading LibriSpeech's 100-hour train-clean-100 set remained an issue locally and on Colab, and was later circumvented by downloading zip files through ASR Corpus website and mounting the drive on Colab. Preprocessing errors also made it harder, with a single flac file going unpaired since its metadata was damaged, which we repaired with validation scripts. Metric computation also caused issues, with the initial WER computations producing false values (e.g., 100%), which we resolved by using proper jiwer computations that accommodate empty predictions and mismatches. Google Colab A100 GPU sessions timed out too frequently, and also incorrectly cached GPU memory while training, causing out of memory errors, requiring reruns, and disrupting training, but checkpointing every 100 steps solved this. Streamlit application deployment was marred by 404 Not Found and 400 Bad Request errors, resolved by correcting endpoint paths and input formats. Heteroscedasticity (speaker and context error variance) and double descent analyses, suggested in Milestone 1 feedback, introduced computational cost but were essential to the evaluation of model robustness. Additionally, careful balancing of context shuffling for SQuAD and noise injection for LibriSpeech was required to enable out-of-distribution (OOD) robustness without overfitting. These problems, which were resolved through incremental debugging, testified to the complexity of the project and impacted scalability strategies.

6 RESULTS

We conducted experiments to test our chatbot's performance in audio transcription and answering questions with the Whisper Tiny model for audio transcription and the Llama 3.2 1B model for question-answering, and testing the Streamlit app we built to bring it all together. The experiments compared the models before and after fine-tuning them on small portions of the LibriSpeech dataset (3 hours of audio) and SQuAD 2.0 dataset (2,500 questions). We also put the app through a longer recording to check its transcription and question-answering functionality. The results, which are in the tables below, check the accuracy with which the models transcribed audio and answered questions, and how well the app fared in real use. All the tests were run on limited computational resources (Google Colab's A100 GPU for models, Windows 11 with Intel Core 9 Ultra and 32 GB RAM for the app).

6.1 Experiments Performed

Transcription Tests: We tested the untrained Whisper Tiny model on a 0.6-hour chunk of LibriSpeech audio to see how accurately it transcribed spoken words. After fine-tuning the model, we ran the same test to check for improvements. We also added slight noise to the audio to see how the model handled trickier conditions.

Question-Answering Tests: Five SQuAD questions were used to test the untrained Llama 3.2 1B model first, and then again after fine-tuning. To test the adaptability of the approach and make the questions more challenging, we randomized the context texts.

Tests of the Streamlit App: As a major deliverable, we developed a Streamlit app that combined the question-answering and transcribing models. The app includes a simple login system (credentials in a local SQLite database), session management (more than one transcription session without logout), audio upload with pagination for files over 30 seconds long, text question input, and answer display. The frontend is Streamlit for usability, and the backend is a Flask script for transcription and answer generation. We tested the app on a Windows 11 machine (32 GB RAM, Intel Core 9 Ultra) with an 11 MB, 15-minute audio file, performing 15–20 test runs to estimate transcription time, question-answering time, and quality of answers.

6.2 Results Obtained

Transcription Results: Table 1 shows Whisper Tiny's performance. The fine-tuned model made fewer mistakes than the untrained one on regular audio and was slightly faster. It also did better with noisy audio, showing some improvement in handling tough conditions. The model was more consistent across different speakers after fine-tuning.

Question-Answering Results: Table 2 shows Llama 3.2 1B's performance. The fine-tuned model improved in choosing better

Table 1. Whisper Tiny Transcription Results

Metric	Base R	FT R	Base N	FT N
WER (%)	19.10	18.41	21.98	20.86
CER (%)	5.68	5.57	7.91	7.63
SeMaScore	0.879	0.880	0.852	0.852
RTF	0.028	0.026	0.030	0.029
Time (sec)	0.154	0.141	0.186	0.173

words for answers and seemed more confident, but it made slightly more mistakes on getting answers exactly right. It struggled with questions where the context was mixed up, suggesting it wasn't as adaptable as we hoped.

Table 2. Llama 3.2 1B QA Results

Metric	Base R	FT R	Base J	FT J
F1 Score	65.00	61.33	56.00	49.43
Exact Match	0.400	0.400	0.400	0.200
BLEU	0.134	0.248	0.115	0.061
ROUGE-1	0.650	0.594	0.560	0.483
ROUGE-L	0.650	0.594	0.560	0.483
METEOR	0.409	0.407	0.386	0.258
Perplexity	8.966	6.488	10.553	7.651

Streamlit App Results: The application could transcribe a 15-minute audio recording within 3–5 minutes averaged over 15–20 test attempts. It responded to questions within 150–400 seconds at an average of 210 seconds. The answers were mostly accurate and contextually appropriate, but the model at times did not capture the emotional tone of questions. The login within the app runs smoothly, allowing users to create new sessions without having to log out, and the SQLite database stores all the transcriptions and answers consistently. Pagination also worked nicely with long audio files, segmenting them into 30-second pieces for processing (Whisper Tiny's limitation). The user interface was clean, with straightforward buttons for typing questions and uploading audio.

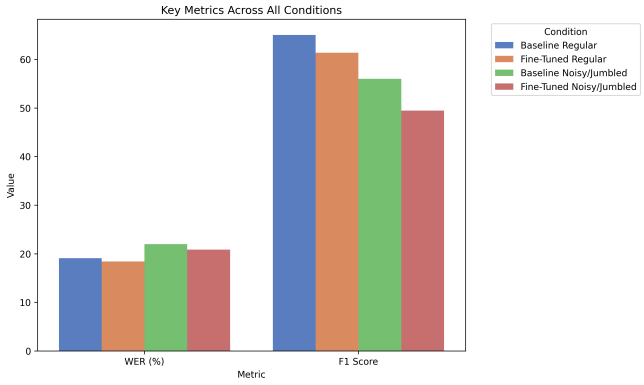

Fig. 1. Evaluation Metrics Comparison

6.3 Summary

The fine-tuned Whisper Tiny model transcribed audio more accurately and quickly than the untrained version, though it could do better with noisy audio. The fine-tuned Llama 3.2 1B model gave better-worded answers but had trouble with mixed-up contexts, showing the limits of our small dataset. The Streamlit app was a big success, making it easy to transcribe long audio files and get relevant answers, though it sometimes struggled with emotional nuances. These results show that fine-tuning helped, but using more data could make the chatbot even better.

7 CONCLUSION AND FUTURE SCOPE

The development and evaluation of our text-based chatbot demonstrated promising results in integrating automatic speech recognition (ASR) and large language model (LLM) capabilities. Fine-tuning Whisper Tiny on a 3-hour LibriSpeech subset reduced transcription errors (WER from 19.10% to 18.41%) and improved speed (latency from 0.154 to 0.141 seconds), while Llama 3.2 1B, fine-tuned on a 2,500-instance SQuAD 2.0 subset, enhanced answer wording (BLEU from 0.134 to 0.248) and confidence (perplexity from 8.966 to 6.488), though F1 slightly declined (65.00 to 61.33). The Streamlit app, a key achievement, seamlessly combined these models, transcribing a 15-minute audio file in 3–5 minutes and answering questions in ~210 seconds on average, with mostly accurate and relevant responses despite occasional struggles with emotional tones. These results highlight the effectiveness of fine-tuning on limited datasets and the app's user-friendly interface, built with login, session management, audio pagination, and SQLite storage, tested robustly on a Windows 11 machine.

The app's success showed that practical deployment is achievable, but transcription times and emotional understanding need refinement. Future work should leverage advanced infrastructure like AWS SageMaker to train on full LibriSpeech (1,000 hours) and SQuAD (130,319 instances) datasets, using stronger models such as Whisper Medium or Llama 3.2 3B to boost accuracy and robustness. Deploying the app on AWS Elastic Beanstalk with S3 storage and Lambda functions could enable real-time transcription and low-latency access for global users. Adding advanced audio augmentation and context-aware training could further enhance emotional tone detection, making the chatbot more versatile for real-world applications.

8 AUTHOR CONTRIBUTIONS

Table 3. Author Contributions

Author	Sections
Yash Mathur	Introduction (10%), Data Description (15%), Conclusion (5%), LLM Finetuning, Documentation
Varun Singh	Approach (15%), Results (10%), Conclusion (5%), ASR Finetuning, App Development
Vishwas Reddy Mali	Literature Review (20%), Conclusion (5%), Writing & Formatting (5%)

REFERENCES

[1] F.-J. Chang et al. 2021. Context-Aware Transformer Transducer for Speech Recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 503–510. https://doi.org/10.1109/ASRU51503.2021.9687895

[2] B. Guan, J. Cao, X. Wang, Z. Wang, M. Sui, and Z. Wang. 2024. Integrated Method of Deep Learning and Large Language Model in Speech Recognition. In *2024 IEEE 7th International Conference on Electronic Information and Communication Technology (ICEICT)*. IEEE, 487–490. https://doi.org/10.1109/ICEICT61637.2024.10671048

[3] Y. Gupta, L. Miao, L. Neves, and F. Metze. 2017. Visual features for context-aware speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5020–5024. https://doi.org/10.1109/ICASSP.2017.7953112

[4] M. Han et al. 2022. Improving End-to-End Contextual Speech Recognition with Fine-Grained Contextual Knowledge Selection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 8532–8536. https://doi.org/10.1109/ICASSP43922.2022.9747101

[5] C. Hori et al. 2019. End-to-end Audio Visual Scene-aware Dialog Using Multimodal Attention-based Video Features. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2352–2356. https://doi.org/10.1109/ICASSP.2019.8682583

[6] M. Kim, G. Kim, S.-W. Lee, and J.-W. Ha. 2021. St-Bert: Cross-Modal Language Model Pre-Training for End-to-End Spoken Language Understanding. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7478–7482. https://doi.org/10.1109/ICASSP39728.2021.9414558

[7] C.-C. Kuo, K.-Y. Chen, and S.-B. Luo. 2021. Audio-Aware Spoken Multiple-Choice Question Answering With Pre-Trained Language Models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), 3170–3179. https://doi.org/10.1109/TASLP.2021.3120638

[8] E. Lakomkin, C. Wu, Y. Fathullah, O. Kalinli, M. L. Seltzer, and C. Fuegen. 2024. End-to-End Speech Recognition Contextualization with Large Language Models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 12406–12410. https://doi.org/10.1109/ICASSP48485.2024.10446898

[9] Junwei Liao, Sefik Eskimez, Liyang Lu, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng. 2023. Improving Readability for Automatic Speech Recognition Transcription. *ACM Transactions on Asian and Low-Resource Language Information Processing* 22, 5 (2023), 1–23. https://doi.org/10.1145/3557894

[10] D. Ortega, C.-Y. Li, G. Vallejo, P. Denisov, and N. T. Vu. 2019. Context-aware Neural-based Dialog Act Classification on Automatically Generated Transcriptions. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7265–7269. https://doi.org/10.1109/ICASSP.2019.8682881

[11] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2103.00020

[12] Y. Sudo, M. Shakeel, Y. Fukumoto, Y. Peng, and S. Watanabe. 2024. Contextualized Automatic Speech Recognition With Attention-Based Bias Phrase Boosted Beam Search. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 10896–10900. https://doi.org/10.1109/ICASSP48485.2024.10447782

[13] Unsloth Team. 2024. LLaMA 3.2 1B Model Card. https://huggingface.co/unsloth/Llama-3.2-1B. Accessed: 2025-04-21.

[14] K. Wei, P. Guo, H. Lv, Z. Tu, and L. Xie. 2021. Context-aware RNNLM Rescoring for Conversational Speech Recognition. In *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 1–5. https://doi.org/10.1109/ISCSLP49672.2021.9362109

[15] W. Xiong et al. 2017. Toward Human Parity in Conversational Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25, 12 (2017), 2410–2423. https://doi.org/10.1109/TASLP.2017.2756440

[16] Imran Zaman. 2024. Fine-tuning 1B LLaMA 3.2: A Comprehensive Article. https://huggingface.co/blog/ImranzamanML/fine-tuning-1b-llama-32-a-comprehensive-article. Accessed: 2025-04-21.

[17] Z. Zhao, Y. Jiang, H. Liu, Y. Wang, and Y. Wang. 2024. LibriSQA: A Novel Dataset and Framework for Spoken Question Answering with Large Language Models. *IEEE Transactions on Artificial Intelligence* (2024). https://doi.org/10.1109/TAI.2024.3427069